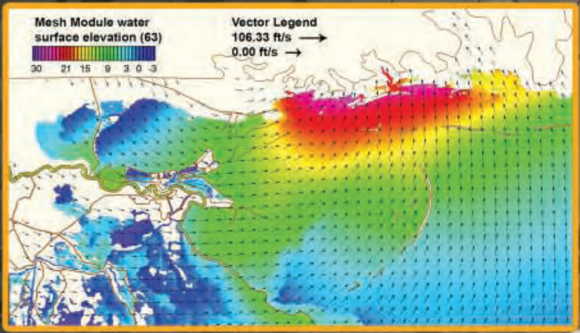
ERDC MSRC

Major Shared Resource Center

RESOURCE



HIGH PERFORMANCE COMPUTING FOR THE WARFIGHTER. SPRING 2006

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **2006**	2. REPORT TYPE	3. DATES COVERED **00-00-2006 to 00-00-2006**
4. TITLE AND SUBTITLE **ERDC MSRC Resource. Spring 2006**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **U.S. Army Engineer Research and Development Center,ATTN: ERDC MSRC HPC Resource Center,3909 Halls Ferry Road,Vicksburg,MS,39180-6199**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**36**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

from the director . . .

John West
Director, ERDC MSRC

This edition of the *Resource* focuses on two issues that have become driving forces during my time here at the U.S. Army Engineer Research and Development Center Major Shared Resource Center (ERDC MSRC): provision of world-class capability and expansion of access to high performance computing (HPC).

I was recently asked to quantify my thoughts on the importance of HPC. I have lots of them, but in terms of really communicating the significance of HPC to the Nation, Dan Reed beat me to the punch during his Congressional testimony in 2004. The essence of what he said is that HPC is distinguished from most other tools of scientific inquiry by its near-universal applicability as an intellectual amplifier. Investments in HPC, unlike investments in astronomical or medical instrumentation for example, stand to benefit the entire research community as a whole, so there is a tremendous force multiplier when adding to the national HPC infrastructure.

Our focus at the ERDC MSRC has been on providing capability for the largest science and engineering problems in the Department of Defense (DoD) while ensuring availability for the "bread and butter" mission in smaller scale problems as well. The most recent machine to enter production at ERDC is the Cray XT3. With over 4,000 processors delivering nearly 22 TFLOPS, the ERDC XT3 is currently the largest publicly announced supercomputer in the DoD and the 14th largest supercomputer in the world. The scale of this machine allows us to simultaneously serve capability and capacity users for the first time, and it has already accomplished a series of "firsts," which you can read more about in this edition of the *Resource.*

There are many components to improving HPC access, and they include some of the hardest problems facing high performance technical computing today: expression of parallel work, effective use of large-scale resources, and performance portability, along with many others. But there are also some more manageable problems that need to be addressed, and we are focusing right now on the HPC interface.

My number one development priorities right now are focused on making HPC more accessible to a larger number of users and scientific communities. In HPC we are using human-computer interaction metaphors that haven't changed substantially since the 1970s. For traditional HPC communities and long-time HPC users, this is not seen as a barrier—it's simply the way it is. But for younger users who have only grown up using a graphical user interface (a substantial and growing proportion of the current science and engineering workforce according to the National Science Board's *Science and Engineering Indicators 2006*) and for communities that have not traditionally used supercomputers, this interaction metaphor is a significant and unnecessary barrier to entry.

I believe deeply in the power of research and development to fundamentally change society for the better, and I believe in the power of HPC to enable new and more far-reaching discoveries. Every dollar we spend right now in making HPC more accessible to new communities and younger users is a dollar that could lead to the next fundamental shift in how we all work and live.

As always, I want to hear from you! If you'd like to let me know how we're doing, share a success story, or make a suggestion for ways to improve our service, drop me a line at **john.e.west@erdc.usace.army.mil**.

John

John E. West, Director
Major Shared Resource Center
U.S. Army Engineer Research and Development Center
Vicksburg, MS

About the Cover: Cover designs by the ERDC MSRC Scientific Visualization Center. For related stories, see pages 2 and 6 (front cover) and page 24 (back cover).

Contents

Hurricane Katrina Researchers Tap High Performance Computing Resources

By Wayne Stroupe, Interagency Performance Evaluation Task Force Public Affairs

Engineering questions surrounding Hurricane Katrina, one of the Nation's worst disasters, are being answered by a dedicated team of experts using some of the most advanced research tools available, including high performance computing assets at the ERDC MSRC.

The Interagency Performance Evaluation Task Force (IPET) was established by the Chief of the U.S.Army Corps of Engineers after Katrina devastated southeast Louisiana and coastal Mississippi on August 29, 2005, and sanctioned by the Secretary of Defense in a directive to the Secretary of the Army on October 19, 2005. IPET's team includes more than 150 nationally and internationally recognized experts from various government agencies, universities, and industry, representing more than 50 organizations.

IPET was charged to answer five questions related to Katrina's affects on the hurricane protection system in and around the vicinity of New Orleans, which encompasses approximately 350 miles of levees and floodwalls. These questions focus on the System (*what was the status of the protection system on August 29*), the Storm (*what exact forces did Katrina put on the system*), the Performance (*how did the system respond*), the Consequences (*understanding the flooding and the losses*), and the Risk (*what is the risk and reliability of the system on and after June 1, 2006*).

Guest writer Wayne Stroupe is permanently assigned to the U.S. Army Engineer Research and Development Center Public Affairs Office. He deployed September-October 2005 to assist with Katrina recovery operations in Louisiana. In November he was temporarily assigned as the public affairs official for the Interagency Performance Evaluation Task Force for the duration of its mission.

With a deadline of providing all these results by June 1, the start of the 2006 hurricane season, IPET organized into 10 analysis teams that have been moving rapidly forward providing individual data and answers that are required by all the IPET teams to accomplish their critical missions. The IPET teams include data collection and management, geodetic vertical and water level datum, hurricane surge and waves, hydrodynamic forces, geotechnical structure performance, floodwall and levee performance, pumping station performance, interior drainage/flooding, consequences, and risk and reliability.

These IPET teams are pushing the engineering envelope with their modeling efforts. The models include traditional modeling activities such as a 1:50-scale physical wave action model of the 17th Street Canal area in New Orleans that is about a third of a football field in size. Other models of levees, using soils from New Orleans, are being tested in complex models that spin on huge research centrifuges located at ERDC in Vicksburg, Mississippi, and at the Rensselaer Polytechnic Institute in Troy, New York.

Other IPET teams are using the powerful capabilities of the ERDC MSRC machines in Vicksburg, including the new Cray XT3.

IPET researchers modeling Katrina's storm surge and the risk of the system for future storms are especially thankful that high performance computing assets were available for their complex runs and use.

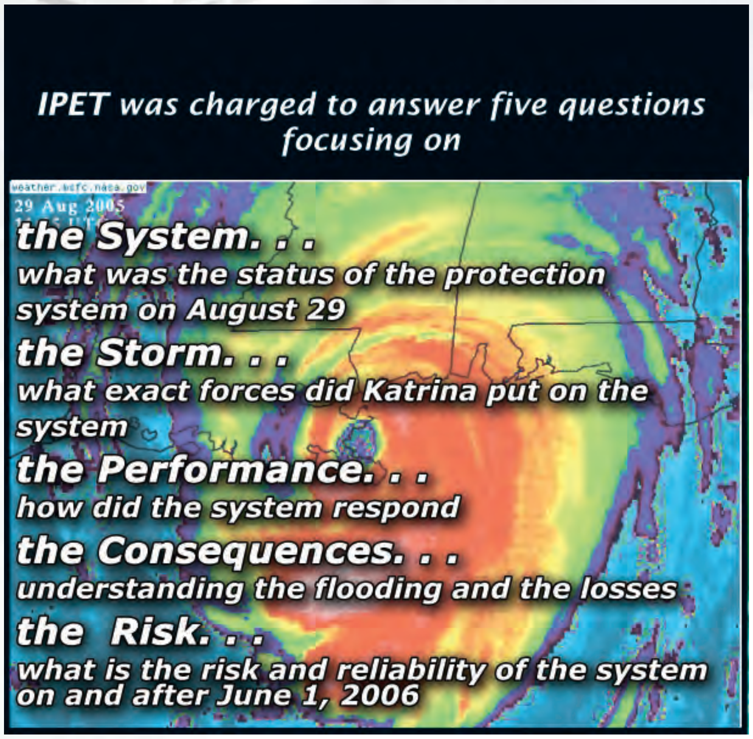
Storm Surge

The IPET hurricane surge and wave analysis team is co-led by Bruce Ebersole, ERDC Coastal and Hydraulics Laboratory, and Joannes Westerink, University of Notre Dame. Westerink is the codeveloper of the Advanced Circulation (ADCIRC) model that the team is using extensively to determine the storm surge during Katrina.ADCIRC is a two-dimensional program that is widely used to determine water surface elevations, velocity, and directional flow in coastal and ocean areas.

For the IPET study, this team is characterizing storm wave and water level conditions along the entire periphery of the hurricane protection system. For the storm surge water levels, the modelers use a base case TF01 grid that includes winds, tides, and Mississippi River flows, and wave input. Each model run involves 377,815 computational points solved every 1 second for 6 days. On the Cray XT3, using 256 processors, the computation takes 75 wall clock minutes.

"The availability of the new CRAY HPC system enabled us to simulate coupling between hurricane storm surge and waves throughout the region at an unprecedented level of detail and technical rigor," Ebersole said.

The animation of the model runs dramatically show how Katrina pushed water up into the coastal areas for days prior to landfall, how the water levels interacted


IPET was charged to answer five questions focusing on

the System. . .
what was the status of the protection system on August 29

the Storm. . .
what exact forces did Katrina put on the system

the Performance. . .
how did the system respond

the Consequences. . .
understanding the flooding and the losses

the Risk. . .
what is the risk and reliability of the system on and after June 1, 2006

weather.msfc.nasa.gov
29 Aug 2005

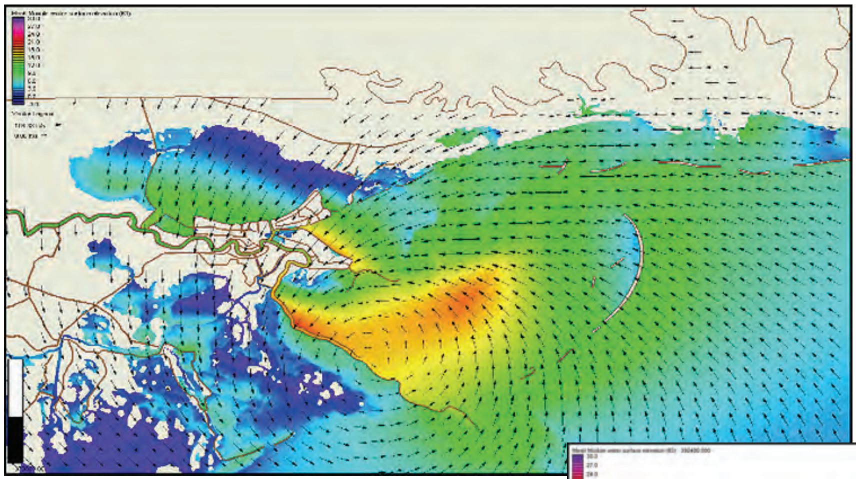
with the coastal land areas and protective levees in specific locations, and how the water levels (especially in Lake Pontchartrain) shifted as the winds shifted during the storm's track.

Coupled with wave model results, these modeling efforts give great insight into how the hurricane protection system performed in specific areas. From the model runs, the levee design heights in and around Plaquemines and St. Bernard's parishes were overwhelmed by the storm by as much as 5 to 6 feet. The model results are also providing great input needed by other IPET teams, such as those studying the 17th Street and London Avenue canal failures.

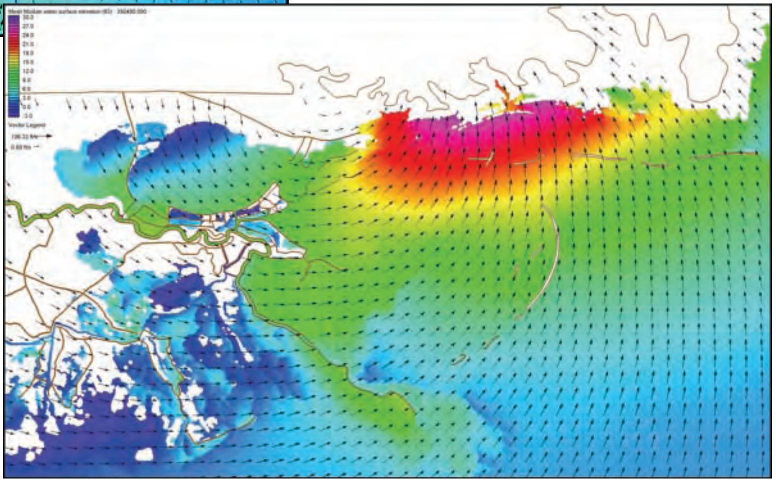
Risk and Reliability

The IPET risk and reliability analysis team is answering the question – "Following repairs to the Katrina-damaged levees on June 1, 2006, what will be the quantifiable risk to the New Orleans and vicinity from future hurricanes and tropical storms?"

The complex answers to this oversimplified question are probably some of the most



ADCIRC model scene showing eye of Katrina just crossing Mississippi River. The different colors relate to the storm surge water levels



ADCIRC model scene showing Katrina hitting Mississippi coast with a storm surge of almost 30 feet

The IPET Risk and Reliability analysis will give the decision makers information on how the hurricane protection system will perform after repairs are completed on the Katrina-damaged sections on June 1, 2006 important results that will come out of the entire IPET study. This team, led by Jerry Foster from Corps Headquarters and Bruce Muller from the Bureau of Reclamation, will provide decision tools that will be critical for residents of New Orleans and decision makers at all levels of government working with the protection system.

This team is using data from all of the other nine IPET teams in their massive effort to quantify the risks to life and property from a range of possible hurricanes and tropical storms. They are using information on the various basins and subbasins around New Orleans, the pumping stations and drainage systems, levee repairs and possible upgrades, system assessments, and other information for the system as a whole and as subunits or components.

These complex system characteristics are coupled with hurricane models that input such storm variables as potential rainfall, track, direction of travel, speed of travel, wind speed, barometric pressure, etc., so that more than 1,800 storm scenarios will be run on the high performance computing (HPC) assets at the DoD ERDC MSRC. These runs are using multiple processors and large amounts of run time.

"The reduced run time made possible by the Cray XT3 has allowed the risk team to examine the full range of hurricanes expected in the New Orleans area. These runs are using multiple processors and large amounts of run time and could not have been completed within the IPET schedule without the new XT3," Foster said.

The risk and reliability analysis will probably transition to other Corps organizations beyond IPET's terminal date of June 1. This work will provide decision makers soon (at the completion of IPET), and potentially in the future, the vital information on where to focus resources on the protection system where a failure would pose the greatest threat to lives and property, where threats will exist once permanent repairs and even authorized components are completed, and where more robust analyses are required. These are the important tools the decision makers in the short- and long-term will need to make critical plans on protecting southeast Louisiana that will ultimately affect repopulating this devastated area.

IPET Final

The Corps' Task Force Guardian that is repairing New Orleans levees to "pre-Katrina" levels by June 1 has been receiving IPET recommendations from the start to ensure the ongoing repairs make optimum use of IPET "lessons learned" so the system is stronger than before. IPET guidance will also be incorporated into future design guidance so that problems discovered by IPET will be corrected in future protection designs and projects.

IPET will issue its draft final report on June 1. All IPET reports to date (January 10 and March 10) are available from the IPET public Web site at *https://ipet.wes.army.mil*, which also has hundreds of other documents related to the hurricane protection system design and construction, IPET data collection, and IPET analyses. All IPET findings and reports are being reviewed and validated by an independent panel from the American Society of Civil Engineers (ASCE). The IPET and ASCE findings are in turn being reviewed and synthesized by an independent panel from the National Research Council (NRC), which should produce its final report in September 2006. IPET will address the final comments by the ASCE and NRC panels and finalize the IPET report in the fall.

Governor Haley Barbour Dedicates Most Powerful Supercomputer in Department of Defense

By Rose J. Dykes, ERDC MSRC

On Wednesday, May 31, 2006, the Honorable Haley Barbour, Governor, State of Mississippi, was the Keynote speaker for the Dedication and Unveiling Ceremony of the new Cray XT3, the most powerful publicly announced supercomputer in the Department of Defense. The brilliance of this gem, named Sapphire, recently joined the array of other supercomputer gems at the ERDC MSRC, elevating the computational capacity to a dazzling 26 TFLOPS (trillion floating-point operations per second). Although Sapphire with its 22 peak TFLOPS of computational capacity outshines the other gems, the Cray X1, named Diamond, the SGI Origin Complex, named Ruby, and the HP/Compaq SC45, named Emerald, continue to add their unique sparkle to the ERDC MSRC computer infrastructure.

ERDC invited the most prominent citizen in Mississippi to unveil the Cray XT3—the Governor. The Honorable Haley Barbour graciously accepted the invitation for the special occasion, presented the Keynote Address at the Dedication Ceremony, unveiled the Crown Jewel of the DoD supercomputers, and even autographed the colossal gem.

"Mississippi ranks among the Top 10 States in supercomputing power in the Nation."
– Gov. Barbour

Governor Barbour praised the work of the ERDC, the U.S. Army Corps of Engineers, the DoD High Performance Computing Modernization Program (HPCMP), and especially that of the ERDC MSRC. He revealed his awareness of the significance of supercomputing in Mississippi when he said, "Mississippi ranks among the Top 10 States in supercomputing power in the Nation."

The Governor also said, "ERDC is a jewel for the State of Mississippi and we're very proud of its role in supporting the Corps and the entire military in the warfighting effort. The addition of the Cray XT3 greatly enhances our position as a technologically advanced State and brings us greater recognition in the scientific community; this is important in terms of luring high-tech economic development as well making us a stronger player in the defense industry. Mississippi supplies

(Continued on page 31)

Dr. Jeffery P. Holland, Director, ERDC Information Technology Laboratory, serves as Master of Ceremonies for the Cray XT3 Dedication Ceremony, May 31, 2006

Dr. James Houston, Director, ERDC, speaks at Ceremony and introduces the Governor of Mississippi

The Honorable Haley Barbour presents Keynote Address

ERDC Cray XT3 Proves Vital in the Certification of the Miniature Air-Launched Decoy

By Dr. Nathan Prewitt and Phil Bucci, ERDC MSRC

The availability of the ERDC Cray XT3 and the requirement for a highly parallel system to provide numerical simulation data for store certification proved to be perfect timing.

The Chief of Staff for the Air Force Special Interest Item Program, which has minimal in-house computing resources, requested time on a massively parallel supercomputer from the High Performance Computing Modernization Program Office (HPCMPO). The HPCMPO offered up the ERDC MSRC Cray XT3 (Sapphire) and the Army Research Laboratory (ARL) Linux Cluster (JVN) to assist the project, which is being led by the Air Force SEEK EAGLE Office (AFSEO) Computational Aeromechanics Team (CAT) at Eglin Air Force Base.

The Miniature Air-Launched Decoy (MALD), shown in Figure 1, is a 250-pound turbojet-powered vehicle that can carry electronic packages allowing it to mimic the radar signature of a wide range of aircraft. It was designed by Raytheon Missile Systems of Tuscon, Arizona, and is expected to enter service in 2008. The MALD vehicles are expected to be low cost, expendable, and typically used in a preplanning role. They will be used to stimulate, decoy, and saturate an Integrated Air Defense System (IADS) during Suppression of Enemy Air Defense (SEAD) missions. If a MALD is tracked or engaged or if it confuses the Command, Control, and Communication (C3) system, it has successfully completed its mission, whether or not it is shot down.

The MALD is designed for unpowered separation from the F-16 with multiple pylon configurations and for powered separation from the airborne electronic attack variant of the B-52H, shown in Figure 2, using the Heavy Storage Adapter Beam (HSAB). The aft section of the MALD includes a unique grid stability augmentation device (SAD) to maintain launch stability during separation. Once clear of the carriage aircraft, the SAD is ejected and the engine ignited.

Placing a weapon on an aircraft can affect its aerodynamic loads, performance characteristics, and flight handling characteristics, etc. Therefore, before a new weapon can be placed on an aircraft for carriage or separation, it must go through a certification process to ensure the safety of all personnel and to maintain the integrity of all military assets. The certification process includes several engineering analyses, many of which require aerodynamic data. These aerodynamic data can be supplied through ground testing, flight testing, or computational simulation such as computational fluid dynamics (CFD). The AFSEO was created as a single point of expertise for Air Force aircraft-store compatibility and certifies all weapons to be placed on Air Force aircraft.

In the case of a weapon being released from an aircraft, the possible trajectories of the weapon immediately after release must be analyzed. There are three methods that have been used to perform this analysis. Captive Trajectory System (CTS) involves a wind tunnel test in which the sting-mounted weapon is placed in the carriage position under the aircraft. Forces and moments are measured on the weapon, and the weapon is repositioned in near real-time. This same process can be simulated using a CFD code combined with a motion simulation. However, in the case of the CFD simulation, time accuracy can be achieved. In both cases, a single trajectory is calculated for each set of initial conditions (i.e., freestream velocity, orientation, etc.). The third form of analysis involves the use of a wind tunnel test to collect an array of aerodynamics data using the weapon in multiple positions under the aircraft. These data are then used to calculate multiple

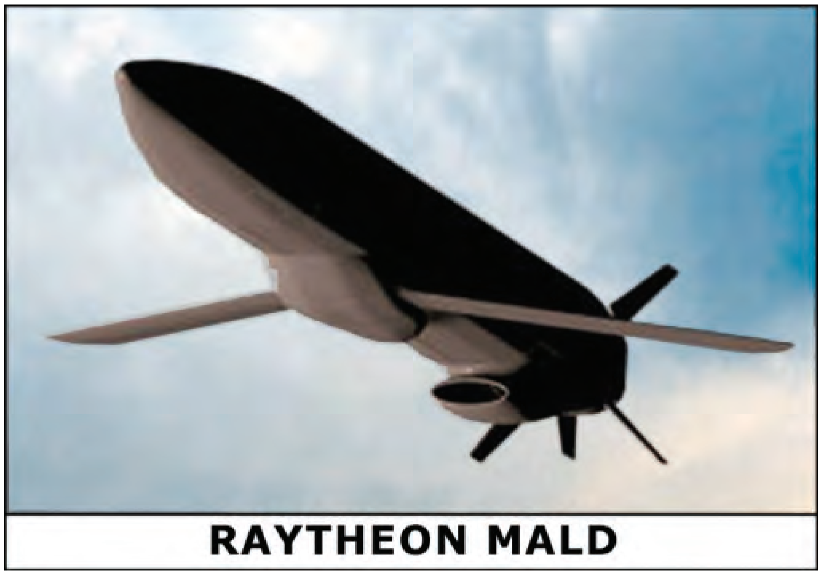


Figure 1. MALD concept image (image by Greg Gobel)

trajectories by interpolating forces and moments from the aerodynamics data at known positions. This process allows running multiple trajectories after the aerodynamics data are collected and some of the mutual interference effects between the weapon and the aircraft are captured. However, it uses interpolated data, and the time evolution of the flow field is lost.

Figure 2. B-52H with concept wingtip jamming pods (image by Guy Aceto)

AFSEO's CFD analysis of the MALD was initiated in late 2004, during the final stages of the design, to study its separation characteristics. The goal was to use CFD to computationally validate safe and effective separation throughout the flight envelope and to support upcoming wind tunnel tests. To perform this form of analysis, the AFSEO CAT team employs a code called Beggar. Beggar is a CFD code that solves the time-accurate fluid flow equations using overset/Chimera and block-to-block, structured meshes. The code was designed for store separation analysis and includes a tightly integrated 6 degree-of-freedom model for the calculation of trajectories of rigid bodies with multiple components.

If the aerodynamics data are to be collected through wind tunnel testing, the B-52H and the MALD have to be scaled down to avoid interference effects with the wind tunnel walls. Since the MALD is so small in comparison with the B-52H, the scaled version of the MALD would present significant challenges to collecting accurate data. Therefore, the MALD program office decided to rely entirely on CFD for the certification of the MALD on the B-52H. Rather than using CFD to calculate time-accurate trajectories for specific flight conditions, the more conservative approach of duplicating the wind tunnel data was taken. Therefore, in late 2005, the MALD project generated a request for a "numerical wind tunnel" simulation consisting of thousands of individual CFD calculations. In interfacing with the program representatives and separation engineers, a minimal number of data points were identified to allow them to perform their evaluations. This resulted in a requirement for 6,000 CFD solutions to be provided by the AFSEO CAT team. Delivery of the solution results would duplicate the aerodynamic data that would be produced by wind tunnel testing but at a substantial savings in cost and time.

AFSEO's in-house computational resources could complete approximately 90 CFD solutions per week. Based on this rate, only about 1,700 of the required 6,000 solutions could be done before the requested deadline. Although the actual central processing unit (CPU) time requirements for a specific calculation may vary because of flow conditions, convergence sensitivity, or the degree of oversight required, a baseline estimate of 224 CPU hours per solution results in a requirement for approximately 960,000 CPU hours to generate the remaining 4,300 solutions. Under near ideal conditions, 500 dedicated processors would allow completion within 80 days; 2,000 dedicated processors would allow completion in less than 3 weeks. This supports the need for dedicated access to a high-capability computing resource such as Sapphire.

Since Sapphire is a new HPCMP asset, much work had to be done to ensure that the Beggar code would be ready to run once the machine was delivered and put into production. Therefore, the CAT team was given access to a small test system called Azurite. Azurite provided a chance to port the code and to validate solution results on a system with the same compilers and hardware as the production machine.

After Sapphire passed Effectiveness Level Testing, the CAT team was granted access during a period of capability testing. This allowed the team to move seamlessly into full production mode. However, even with the initial preparation, problems still arose during the production runs. Foremost, the shear size of the mesh systems used for the MALD, B-52H, and carriage hardware, as shown in Figure 3, brought with it a significant memory requirement. This was particularly evident during the process of assembling the overset mesh system. Fortunately, this process only has to be done once for steady-state problems. Therefore, in order to avoid the extensive code modifications that would be required otherwise, the ERDC MSRC

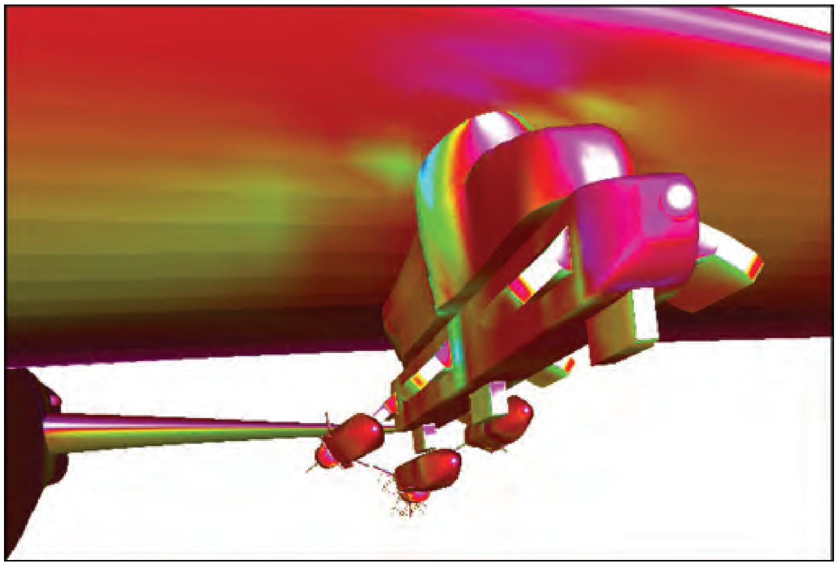


Figure 3. CFD solution of three MALD configuration on B-52H

purchased additional memory for the login nodes so that this part of the solution process could be done as a preprocessing step.

On behalf of the SEEK EAGLE Office, in mail delivered to John West, ERDC MSRC Director, and David Stinson, the ERDC MSRC Operations Manager, Colonel Thomas A. Buter commended ERDC for support of the MALD certification analysis for the B-52H during January through March 2006. He noted that the capabilities made available in the form of troubleshooting support and 1.27 million CPU hours had allowed production of more than 1,000 individual CFD solutions in support of the program. The expertise and enthusiasm of the ERDC MSRC staff was noted as being abundantly evident from the superior support the team provided.

To quote Colonel Buter, "CFD has been employed to generate the large number of individual solution data points typically provided by a wind tunnel support effort. The ERDC MSRC support team has ably supported this project and the validation of CFD's ability to virtually replace wind tunnel testing where needed and where appropriate by providing a comparable amount of data in a comparable or shorter amount of time. The contributions of the ERDC MSRC team have been critical to substantially reduce both risk and cost, while conserving schedule for a critical acquisition program."

CAT Team (bottom row, left to right): Alex Dobrinsky, Jason Torres, Capt. Jacob Freeman, Robert Moran (top row, left to right: Joseph Keen, Bruce Jolly, John Martel, Lt. Judson Babcock, Robert Spinetti, John Fay, Mark Kannapel, Sergey Kernazhitskiy, Magdi Rizk

Nonlinear Response of Materials to Large Electric Fields: A Capabilities Application Project on the Cray XT3

By Dr. James Freericks, Dep artment of Physics, Georgetown University

Nonlinear effects in nature are often the most important and most interesting effects in science. Many electronic devices either have their operation based on their nonlinear characteristics (like the transistor) or have the regime where they stop operating properly determined by where nonlinear effects become too strong (like intermodulation distortion in passive microwave filters). There is much interest, from both a basic science and an applications perspective, in understanding nonlinear effects in strongly correlated materials. This is because strongly correlated materials have many properties that can be tuned by simply applying pressure, changing the temperature, or by chemical doping. Examples of these materials include the high-temperature superconductors, rare-earth magnets, Kondo-effect and heavy Fermion materials, Mott insulators, and so on. Strongly correlated materials are characterized by having a strong enough electron-electron interaction that the independent particle picture (i.e., band theory) does not apply, and their behavior is determined by complex quantum-mechanical dynamics. The most interesting materials are those that lie close to a phase boundary between different types of materials, like the boundary between a metal and an insulator, or between a magnetic and a nonmagnetic system.

As the military becomes more and more "high-tech," it relies on electronics in both the analog and digital realms for all forms of operation. These electronics devices are susceptible to attack or damage from large electric fields (either from natural sources or from high-energy, pulsed-beam weapons). In addition, the military is moving toward "smart" electronics, which have elements that can be tuned "on the fly" to respond to different needs of the user. Strongly correlated materials, because of their inherent high tunability, are solid candidates for use in such devices. Unfortunately, little is known about their response to these kinds of electric fields, and modeling can provide useful answers to help engineer and design the next generation of smart electronics devices.

This Capabilities Application Project (CAP) provides the first numerically exact solution to this long-standing problem.

Strongly correlated materials require the solution of the so-called many-body problem to understand their behavior. For years, the many-body problem was viewed to be essentially intractable, but significant progress has been made over the past two decades as computational algorithms became more sophisticated, and as new ideas emerged for successfully solving the quantum-mechanical problem. Dynamical mean-field theory [1] was introduced in 1989 as a new way to approach this problem. In the ensuing 17 years, it has been employed to solve most of the "classic'" models in many-body physics (Hubbard, periodic Anderson, Falicov-Kimball, Holstein, etc.) [2] in equilibrium. These solutions have provided tremendous insight into the Mott metal-insulator transition and the role of strong electron correlations in real materials.

Our project extended this highly successful approach from the equilibrium case to the nonequilibrium case (in the presence of an external electric field), which allows the nonlinear response of the material to be calculated. The formalism for solving the nonequilibrium many-body problem (in an arbitrary strength external field) was worked out in the early 1960s [3,4]. We solve the simplest many-body model that illustrates this behavior. The model has two kinds of particles on a crystal lattice: mobile electrons, which can move from one lattice site to the nearest neighbor site, and localized electrons, which do not move [5]. When a mobile electron sits on the same site as a conduction electron, there is an energy cost (of magnitude U) because of

Guest writer Dr. James Freericks received his bachelors degree (*summa cum laude*) in Physics from Princeton University in 1985 and his Ph.D. in Physics from the University of California, Berkeley, in 1991. After spending time as a postdoctoral fellow at UC's Santa Barbara and Davis campuses, he went to Georgetown University, where he has been a Professor of Physics since 1994. He has been funded by the Office of Naval Research, starting with a Young Investigator Program Award, since 1996, and has been involved in high performance computing on DoD machines since 2001. This was his first CAP.

the mutual electrical repulsion between the particles. If U is large enough, and the number of mobile plus localized electrons equals the number of lattice sites in the crystal, then *all* electrons become frozen, and the material cannot conduct electricity – the system undergoes a metal-to-insulator transition as a function of U.

In nonequilibrium calculations, we need to work with objects called Green's functions, which describe the distribution in energy of available quantum states in the system, and the statistical occupation of those states as they are driven by the external electric field. Our formulation works in real time, where the transient effects are included, and the steady-state response only builds up after a long period. The Green's functions are continuous matrix operators defined on the so-called Kadanoff-Baym-Keldysh contour, which involves evolving the system forward in time from some starting point to some ending point, and then de-evolving the system backward in time from the ending point to the starting point. One of the numerical challenges of this approach is that we need to discretize the continuous matrix operator so we can perform matrix operations on it via computer-based linear-algebra techniques. We are limited, by computational time (and to some degree by memory), in how large a matrix we can use in the calculations, which ultimately limits how far out in time the calculations can go.

The formalism is straightforward, but complex. The noninteracting solution in arbitrarily large electric field is known [6], so we need to add in the effects of the electron-electron interactions. We use the dynamical mean-field theory approach to do this, which requires

The formalism is straightforward, but complex.

us to solve a set of nonlinear matrix-valued equations self-consistently. Describing the quantum-mechanical basis for this algorithm is beyond the scope of this article. Instead we focus here on the numerical issues and the use of the XT3 for large-scale parallel programs. The most challenging numerical part of the algorithm involves evaluating a two-dimensional Gaussian-weighted matrix-valued integral to determine the Green's function. The integrand requires a matrix inversion and a matrix multiplication to determine its value (the matrix is a general complex matrix). We use Gaussian integration and average the results of a discretization of 100 points per dimension and 101 points per dimension for a total of 20,201 quadrature points, and hence 20,201 matrix inversions and multiplications per iteration step. Since the matrices in the integrand are independent of one another, this part of the algorithm is easily parallelized in the master/slave format, by sending the matrix inversions to different slave nodes. After the integration is completed, the master node needs to perform an additional four matrix inversions to complete one step of the iteration. These calculations must proceed in a serial fashion, because results from the previous calculation are required for the next. Because of this structure, we expect the computational time for the code to be expressed as a sum of these two pieces $T = T_0+T_1/N$, where T_0 is the time for the serial part of the code, T_1 the total time for the parallel part, and N the number of slave nodes during the run. Of course, during the time that the serial

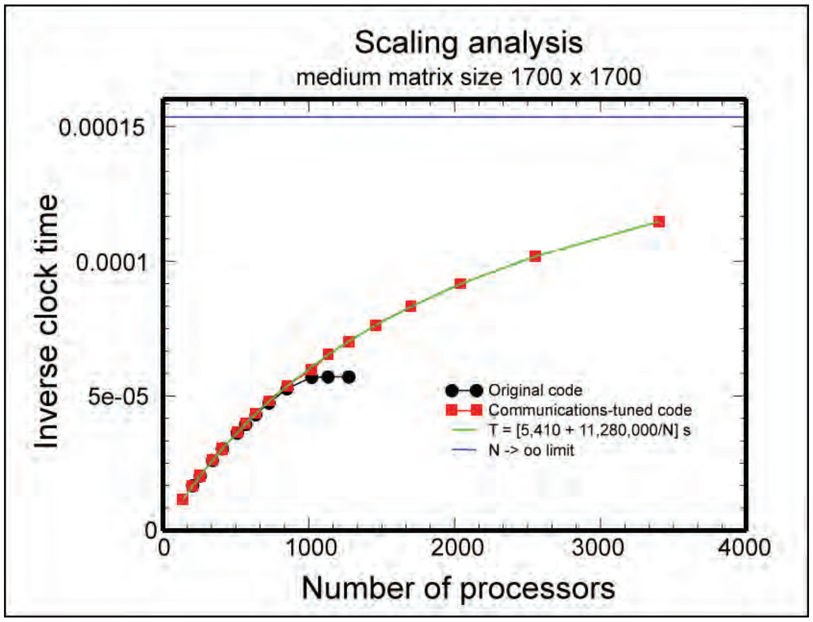


Figure 1. Run time scaling analysis for a medium-sized code on the Cray XT3. The black symbols are the original code; the red are the communications-tuned code. The green is the fit to the expected form because of the serial part of the code. The blue is the limiting result for N = 10,201 processors or more

part of the code is running, the slave nodes are idle, and the relative fraction of time for this part of the code increases as the number of nodes increases. Hence the code cannot display complete linear scale-up with the number of nodes, but we will see that we are able to perform a linear scale-up of the parallel part of the code up to the maximal number of nodes that it makes sense to run on. Since the serial part of the code is at most only about half of the total computational time (and often is much less), we run with an efficiency of 50 percent or more.

The code is quite robust, having been run on Beowulf-type machines, the T3E, and the X1, prior to being ported to the XT3, so it was believed that there would be little required to get the code to scale well to large numbers of slave nodes. This turned out not to be true, as can be seen from our scaling attempts on a medium-sized problem of 1700X1700 matrices, with the black symbols in Figure 1. Once we reached about 1,000 processors, the code stopped scaling with the number of slave nodes. We quickly realized the problem here was a communications bottleneck of using a many-to-one communications method to send the matrix results from each slave node to the master node for accumulation of the quadrature. This problem was easily fixed by using a recursive-binary-gather operation to accumulate results on the nodes, in turn, and then send the final results to the master node. One can see the scaling behavior returns by looking at the red symbols for the communications-tuned code in Figure 1. The recursive gather scheme involves taking all the active slave nodes, dividing them in half, and sending the results from one half of the nodes to the other half. Once a slave node has sent its results, it becomes inactive. The process is repeated until only one active slave node remains. At this point, it sends the accumulated results to the master node, and the serial part of the code starts. The serial code takes the local Green's function matrix (which resulted from the two-dimensional integration) and performs four more matrix inverses to construct the next approximation to the self-energy matrix. The system is checked for convergence, and if not converged, the iterations continue. Typically, we need between 10 and 50 iterations to reach convergence. Once the calculations have converged for the Green's functions, we can directly extract the electron distribution functions. These data tell us how the electrons directly evolve in response to the electric field. In addition, we also investigate the current as a function of time, to see the transient effects of oscillations.

The results of the scaling analysis show that the code runs well on a large number of processors. The rollover as N becomes large occurs because of the serial part of the code. If we follow the initial slope, for the linearly scaling part of the curve, we can see that in virtually all results, the efficiency is better than 50 percent; so choosing an optimal number of processors requires us to balance the efficiency factor versus the actual clock time for the code. On the XT3, we find a medium-sized problem, with 1700X1700 matrices taking about 1,100 hours per iteration, with between 15 and 30 iterations needed for convergence. Increasing the size to 2200X2200 resulted in an increase of the computational time to about 2,000 hours per iteration (because the matrix inversion is the limiting part of the code, the time should scale as the cube of the matrix size). We could not increase the problem size larger than this on the XT3, because the memory per node was just 1GB at the time. Results for different discretization sizes need to be scaled to the limit where the discretization size goes to zero to achieve reliable results. We could show, by comparing to exact sum rules [7], that our results improved in accuracy from an average error of a few percent to less than one-tenth of a percent when we could scale to the zero discretization limit. Our CAP project used about 600,000 CPU hours for the scaling analysis and production runs over a period of about 6 weeks. We are currently preparing these results for publications in peer-reviewed journals.

References:

[1] W. Metzner and D. Vollhardt, "Correlated Lattice Fermions in d=∞ Dimensions," Phys. Rev. Lett. **62**, 324 (1989).

[2] A. Georges, G. Kotliar, W. Krauth, M. J. Rozenberg, "Dynamical mean-field theory of strongly correlated Fermion systems and the limit of infinite dimensions," Rev. Mod. Phys. **68**, 13 (1996).

[3] L. P. Kadanoff and G. Baym, *Quantum Statistical Mechanics*, (New York, Benjamin, 1962).

[4] L. V. Keldysh, "Diagram Technique for Nonequilibrium Processes," J. Exptl. Theoret. Phys. (USSR) **47**, 1515 (1964) [Sov. Phys. JETP **20**, 1018 (1965)].

[5] L. M. Falicov and J. C. Kimball, "Simple model for semiconductor-metal transitions: SmB_6 and transition-metal oxides," Phys. Rev. Lett. **22**, 997 (1969).

[6] V. Turkowski and J. K. Freericks, "Nonlinear response of Bloch electrons in infinite dimensions," Phys. Rev. B **71**, 085104 (2005).

[7] V. O. Turkowski and J. K. Freericks, "Spectral moment sum rules for strongly correlated electrons in time-dependent electric fields," Phys. Rev. B **73,** 075108—1-15 (2006).

Tuning Codes for Performance on the Cray XT3

By Drs. Sam B. Cable and Thomas C. Oppe

Sapphire, the newest HPC platform at the ERDC MSRC, is a distributed-memory massively parallel processor (MPP) Cray XT3. It is comprised of 4,176 single-processor nodes using 2.6-GHz AMD Opteron processors. Of these, 4,128 are "compute" nodes for running jobs, and 48 are "service" nodes that perform support functions for application and system services. Of the service nodes, 10 are login nodes and 23 are input/output (I/O) server nodes for the Lustre file system (i.e., the `/work` directory). The compute nodes run Catamount, a microkernel operating system, while the service nodes run SuSE Linux. On May 15, 2006, the compute nodes were upgraded to 2 GB of memory. The service nodes generally contain 4 GB of memory except for the login nodes, which contain 8 GB of memory. All nodes are connected in a three-dimensional torus using a Hyper-Transport link to a dedicated Cray SeaStar communications subsystem. Sapphire is rated at 22 peak TFLOPS and contains 36 TB of Fiber Channel RAID disk space.

The login nodes are named **sapphire01, sapphire02, …, sapphire10**. To login to **sapphire04**, for example, the user issues one of the following commands:

```
% telnet sapphire04.erdc.hpc.mil
% rlogin sapphire04.erdc.hpc.mil
% ssh sapphire04.erdc.hpc.mil
```

Use of Modules

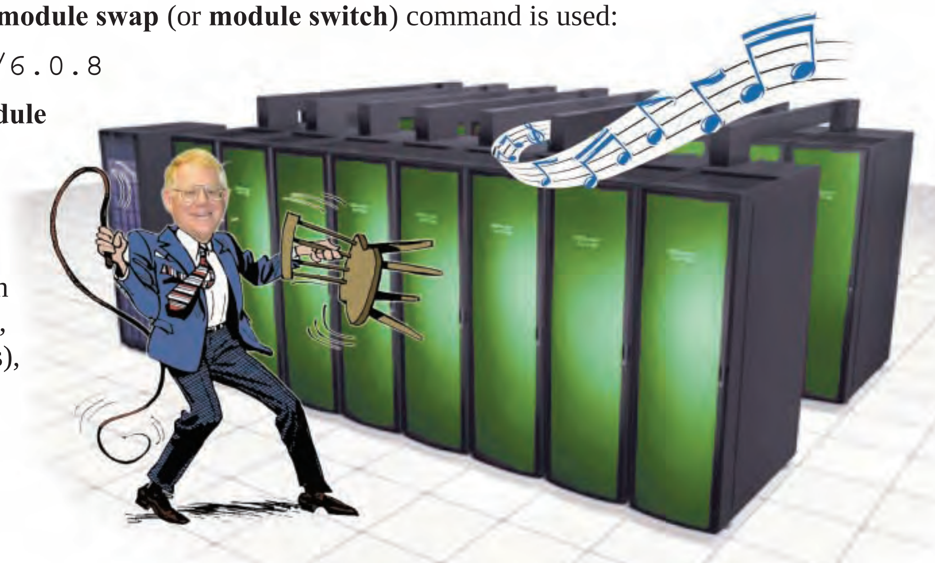
The first step in optimizing codes is selecting a compiler and compiler options. The compilers and libraries on the XT3 are selected using the **module** command. For example, the command **module list** will show the currently loaded versions of the compilers and libraries:

```
% module list
Currently Loaded Modulefiles:
  1)  modules/3.1.6         10) xt-libc/1.3.21
  2)  acml/2.7              11)  xt-os/1.3.21
  3) pgi/6.1.1              12) xt-catamount/1.3.21
  4)  xt-libsci/1.3.21      13) xt-boot/1.3.21
  5)  xt-mpt/1.3.21         14) xt-crms/1.3.21
  6) xt-pe/1.3.21           15) xt-lustre-ss/1.3.21
  7) papi/3.2.1             16) Base-opts/1.3.21
  8)  PrgEnv-pgi/1.3.21     17) x t-lsfhpc/6.1
  9)  xt-service/1.3.21     18) xt-iobuf/1.0.0
```

The Portland Group, Inc. (PGI), compilers, version 6.1.1, are the default at the time of writing. To find the available versions of the compilers and libraries, the user issues the command **module avail**. If the user wishes to load an older compiler, such as version 6.0.8, the **module swap** (or **module switch**) command is used:

```
% module swap pgi/6.1.1 pgi/6.0.8
```

The **module load** (or **module add**) and **module unload** (or **module rm**) can load or unload individual modules if swapping of modules is not needed. The **module show** (or **module display)** commands will show how a module affects the user's environment variables, such as `$PATH` (the search path for executables), `$MANPATH` (the search path for man pages), and `$LD_LIBRARY_PATH` (the search path for libraries):



```
% module show pgi/6.1.1
/opt/modulefiles/pgi/6.1.1:
setenv            PGI_VERSION 6.1
setenv            PGI_VERS_STR 6.1.1
setenv            PGI /opt/pgi/6.1.1
prepend-path      LM_LICENSE_FILE /opt/pgi/6.1.1/license.dat
prepend-path      PATH /opt/pgi/6.1.1/linux86-64/6.1/bin
prepend-path      MANPATH /opt/pgi/6.1.1/linux86-64/6.1/man
prepend-path      LD_LIBRARY_PATH /opt/pgi/6.1.1/linux86-64/6.1/lib
prepend-path      LD_LIBRARY_PATH /opt/pgi/6 .1.1/linux86-64/6.1/libso
```

This command is very useful for locating the directories containing the compiler binaries or vendor libraries such as the AMD Core Math Library (ACML):

```
% module show acml/2.7
/opt/modulefiles/acml/2.7:
setenv            ACML_BASE_DIR /opt/acml/2.7
setenv            ACML_COMPILER pg i
setenv            ACML_TYPE 64
setenv            ACML_DIR /opt/acml/2.7/pgi64
prepend-path      LD_LIBRARY_PATH /opt/acml/2.7/pgi64/lib
prepend-path      PE_PRODUCT_LIST ACML
```

Thus the directory /opt/acml/2.7/pgi64/lib contains the ACML (version 2.7) library for PGI compilers using 64-bit pointers.

Optimization via Compiler Options

The names of the Portland Group FORTRAN and C compilers are **pgf90** and **pgcc**, respectively, for serial programs and **mpif90** and **mpicc** for parallel MPI programs.

Important options for these compilers are:

`-V`	displays the version number of the compiler
`-fastsse`	most aggressive optimization
`-fast`	moderate optimization
`-Mscalarsse`	use fast SSE and SSE2 instructions for 64-bit floating point
`-tp k8-64`	target binary for the AMD Opteron in 64-bit mode, default
`-tp amd64`	same as the previous option
`-byteswapio`	swap bytes from big-endian to little-endian or vice versa on input or output of binary FORTRAN data files
`-Mfreeform`	FORTRAN source files are in free format style
`-Mipa=fast`	chooses generally optimal InterProcedural Analysis flags
`-Kieee`	perform floating-point operations in strict conformance with the IEEE 754 standard. Some optimizations are disabled, and a slightly more accurate math library is used. `-Knoieee` is the default.
`-O0`	no optimization
`-O1`	low optimization
`-O2`	moderate optimization
`-O3`	high optimization
`-O`	same as `-O2`

The **–fast**, **–fastsse,** and **–Mipa=fast** options are conglomerations of other optimization flags. The individual optimization components of these options can be viewed by invoking **pgf90** or **pgcc** with the option and an additional **–help** flag, as in

```
% pgf90 -fastsse -help
```

The man pages for **pgf90** or **pgcc** can then be consulted for the definitions of these subsidiary options.

A general procedure for tuning the performance of a code with compiler options is given in the steps below.

Step 1: Compile all files with the most aggressive optimization flag **–fastsse**. If the compilation fails or the application does not run properly, use **–fast –Mscalarsse** instead. If problems persist, use **–O1** or **–O0** instead.

Step 2: Profile the execution of the binary using a profiler such as **gprof** (the GNU profiler), **pgprof** (the Portland Group profiler), or CrayPAT (the Cray profiler). An example of the use of CrayPAT is given below. Determine which routines are critical for performance.

Step 3: Repeat step 1 for the routines critical for performance one at a time, and repeat the run after each recompilation to check for correct output.

Optimization via Source Code Modifications

Tuning the performance of a code by using compiler options alone is not always possible. Often code modifications are necessary. It is a truism that code optimization begins with scalar optimization, i.e., mapping the algorithm to the scalar processor. The AMD Opteron, like all modern RISC-based chips, has functional registers and a hierarchy of caches called L1 and L2. The registers store operands for the functional units. The L1 cache is the smaller of the two caches and is used for storing instructions and data for the registers. The larger L2 cache transfers data between the L1 cache and main memory. On the Opteron, the L2 data cache is 1 MB, and the L1 cache is 128 KB (64 KB each for instructions and data). In order to achieve optimal performance, the user must "program the cache," i.e., reuse data in the caches as much as possible before flushing the cached data to make room for other data from main memory. It is also important to access main memory by contiguous locations if possible since data are transferred between main memory and the L2 cache in "pages" that represent contiguous memory locations. It has been said that the stride of memory references is the single most important issue for performance on cache-based microprocessors. Thus, multidimensional arrays should be accessed by their leftmost index first in FORTRAN codes and by the rightmost index first in C codes.

The user can also take advantage of vendor mathematics/numerical libraries such as the ACML and the Cray XT3 LibSci library. The ACML library contains the LAPACK linear algebra package as well as optimized routines for the Basic Linear Algebra Subprograms (BLAS levels 1, 2, and 3), Fast Fourier Transforms (FFTs), fast scalar, vector, and array math transcendental operations, and random number generators. The Cray XT3 LibSci library includes ScaLAPACK (a parallel version of LAPACK), BLACS (Basic Linear Algebra Communication Subprograms), and SuperLU_DIST (a parallel sparse matrix solver).

As an example of an exercise in code optimization, consider the following FORTRAN program for computing a matrix-vector product $c = Ab$, where A is a full square matrix of order n and b and c are vectors of length n. The mathematical definition of the operation is $c_i = \sum_j a_{i,j}\, b_j,$ which may lead an inexperienced FORTRAN programmer to use the following inner product algorithm:

```
do i = 1, n
  sum = 0.0d0
   do j = 1, n
     sum = sum + a(i, j)*b(j)
   enddo
  c(i)  = sum
enddo
```

For each value of the row index i, the inner loop will access all the elements in row i of the matrix A before going to row $i+1$. However, since FORTRAN arrays are stored in column-major order (i.e., column 1 elements of A are stored first, then column 2 elements, etc.), this algorithm will cause many noncontiguous memory references.

Pages of memory brought into cache that are near element $a_{i,j}$ may well contain only elements in column *j*. Then the reference to $a_{i,j+1}$ in the next iteration of the inner loop will cause a new page containing only elements in column *j+1*. By the time *j* is *n* in the inner loop, any cache pages containing column 1 elements have been flushed to make room for new pages. But at the next iteration of the outer loop, the first column of row *i+1* is referenced, so a new page of column 1 elements must be brought in to the cache. This phenomenon is called cache-thrashing.

To eliminate noncontiguous memory references, an outer product algorithm can be used. The *c* vector is first initialized to zero and then updated by scalar multiples of the columns of *A*.

```
do i = 1, n
  c(i) = 0.0d0
enddo
do j = 1, n
  do i = 1, n
   c(i) = c(i) + a(i, j)*b(j)
  enddo
enddo
```

Note that the outer loop now accesses the columns of *A* in order. Since *A* is stored sequentially by columns, only stride-1 memory references of *A* are made in the inner loop. Pages of memory brought into cache will contain other elements of *A* in the same column, and these elements can be used in later iterations of the inner loop.

A further improvement to the outer product algorithm can be made by observing that the vector *c* is read from memory and written to memory for each iteration of the outer loop. By unrolling the outer loop, it is possible to reduce the memory traffic resulting from accessing the *c* vector. The following code unrolls the *j* loop to a depth of 2 (assume *n* is even):

```
do i = 1, n
  c(i) = 0.0d0
enddo
do j = 1, n, 2
  do i = 1, n
   c(i) = c(i) + a(i, j)*b(j) + a(i, j + 1)*b(j + 1)
  enddo
enddo
```

Assuming that the entire *c* vector and two columns of *A* can fit in cache, this algorithm will halve the memory references to c. One can attempt to unroll the outer loop further, say to a depth of 4 or 8, if more columns of *A* will fit in cache. Clearly, the optimal depth of unrolling will depend on the order *n* of *A* relative to the cache size.

Other approaches are to try the FORTRAN 90 intrinsic function **matmul**:

```
c = matmul (a, b)
```

or the BLAS2 routine DGEMV from ACML for computing $c = \beta c + \alpha Ab$:

```
alpha = 1.0d0
beta = 0.0d0
call dgemv ('N', n, n, alpha, a, n, b, 1, beta, c, 1)
```

In this way, the user takes advantage of the optimized routines available from the compiler or mathematics libraries.

An experiment was conducted on Sapphire to compare times and megaflop rates for each of these methods using 8-byte real-valued operands and a matrix order of 12,000. For the unrolled outer product method, unrolling depths of 2, 3, 4, and 8 were tried. The test program was compiled with the command

```
% pgf90 -fastsse -O3 test.f -lacml
```

and the results were as follows:

Algorithm	Time(sec)	MFLOPS
Inner	7.4010	38.91
Outer	0.5060	569.17
Unroll2	0.3750	768.00
Unroll3	0.3630	793.39
Unroll4	0.3620	795.58
Unroll8	0.3870	744.19
MATMUL	0.3990	721.80
DGEMV	0.3640	791.21

It can be seen that the outer product method is far superior to the inner product method. Apparently, using unrolling to a depth of 3 or 4 in the outer product algorithm maximizes the performance and is roughly equal to the performance of DGEMV.

Another optimization involves using 4-byte real-valued operands instead of 8-byte operands. This technique effectively doubles the size of the L2 cache relative to the size of the operands, thus allowing more data to fit into cache. The corresponding 4-byte BLAS2 routine for computing a matrix-vector product is called SGEMV. The same experiment conducted with 4-byte operands had results as follows:

Algorithm	Time(sec)	MFLOPS
Inner	9.7230	29.62
Outer	0.2120	1358.49
Unroll2	0.1880	1531.92
Unroll3	0.1820	1582.42
Unroll4	0.1800	1600.01
Unroll8	0.1850	1556.75
MATMUL	0.2220	1297.30
SGEMV	0.1800	1600.01

It can be seen that the performance of each algorithm roughly doubles with the exception of the inner product method, which becomes worse. Thus, the user may consider using 4-byte operands as an optimization technique if his program does not require 8-byte precision.

I/O Considerations

There are other considerations for improving code performance. If a code performs significant I/O, the executable and any data files should be placed in a user's `$WORKDIR` directory (i.e., `/work/$USER`), and any jobs should be run from that directory rather than from the user's home directory. The `/u` and `/work` file systems have very different performance characteristics. The `/u` file system is NFS-mounted and is thus tuned for small I/O operations. The `/work` file system, on the other hand, is serviced by the 24 Lustre file system service nodes and is tuned for large I/O operations. In addition, because of certain characteristics of the XT3 hardware and OS, a batch job cannot access the home directories efficiently.

A user's code may also suffer poor performance because of poorly structured I/O. In fact, experience so far indicates that optimal structuring of I/O is especially important on the XT3. A code that writes a large number of small records to disk will not realize its potential performance on the XT3 without code modifications or special steps taken in compiling. Suggestions for improving a code's I/O performance appear in the article on Cray XT3 I/O by Dr. Jeff Hensley and Bobby Hunter elsewhere in this issue.

Automatic Arrays in FORTRAN Codes

A FORTRAN code that makes frequent use of automatic arrays will perform poorly on the compute nodes without special compiling. An automatic array is one that is allocated by a subroutine at run time when the routine is called and deallocated when the routine exits. Usually, the array size is determined by arguments passed to the subroutine. For example, “a” is an automatic array in the following code fragment:

```
      subroutine calculate (m, n)
      dimension a(m, n)
      ...
```

A FORTRAN code that makes frequent calls to subroutines using automatic arrays can benefit from the use of the GNU **gmalloc** library. Source codes changes are not necessary to use this library. The user simply adds the library at the link step. For example:

```
%  pgf90 main.f calculate.f -lgmalloc
```

Users of one code that made extensive use of automatic arrays reported that the use of the GNU library decreased the run time by 15 to 40 percent, with runs made using many processors generally showing the greatest improvement.

Performance Analysis

Any code’s performance can be improved using performance analysis tools. The Cray Performance Analysis Tool, CrayPAT, is the premier analysis tool on the XT3. CrayPAT does not require source code changes; the user first builds his executable as he normally would and then uses CrayPAT to instrument the executable. The user then runs the instrumented executable and examines the resulting output with any of several available tools. The user first loads the **craypat** module:

```
% module load craypat
```

If the user wishes to take advantage of Cray’s graphical tools in analyzing his performance data, he also loads the **apprentice2** module:

```
% module load apprentice2
```

Next, the user makes a typical build of the executable:

```
% mpif90 -o myexec main.f
```

The user then instruments the executable using the **pat_build** utility. **pat_build** requires at least two arguments: the name of the input executable and the name of the output instrumented executable. For example, if the instrumented executable is to be named **myexec_inst**, the appropriate command is

```
% pat_build -u myexec myexec_inst
```

Flags are also available for specifying which particular source routines are to be profiled. See the **pat_build** man page for details.

The user then runs the instrumented code using the **pat_run** utility:

```
% pat_run yod -np  n ./myexec_inst
```

where *n* is the number of processors to be used for the run. **pat_run** will create a subdirectory pat_run.* containing performance data in the form of an ASCII file named myexec_inst.*nn*.xf, where *nn* is a two-digit number assigned sequentially by **pat_run**. The performance data in the file, such as the amount of time the executable spent in various subroutines, can be accessed in a number of ways. The data can be viewed graphically (provided the **apprentice2** module has been loaded) by using the **app2** utility. The user first runs **pat_report** to convert the output into XML format:

```
% pat_report -f xml -c records   myexec_inst.nn.xf
```

Finally, the user runs **app2** to visualize the output:

```
% app2 myexec_inst. nm.xml
```

An example of the graphical output that can be obtained from **apprentice2** is shown in Figure 1.

Acknowledgment

The authors wish to thank Dr. Alan Minga of Cray, Inc., for his helpful comments and advice.

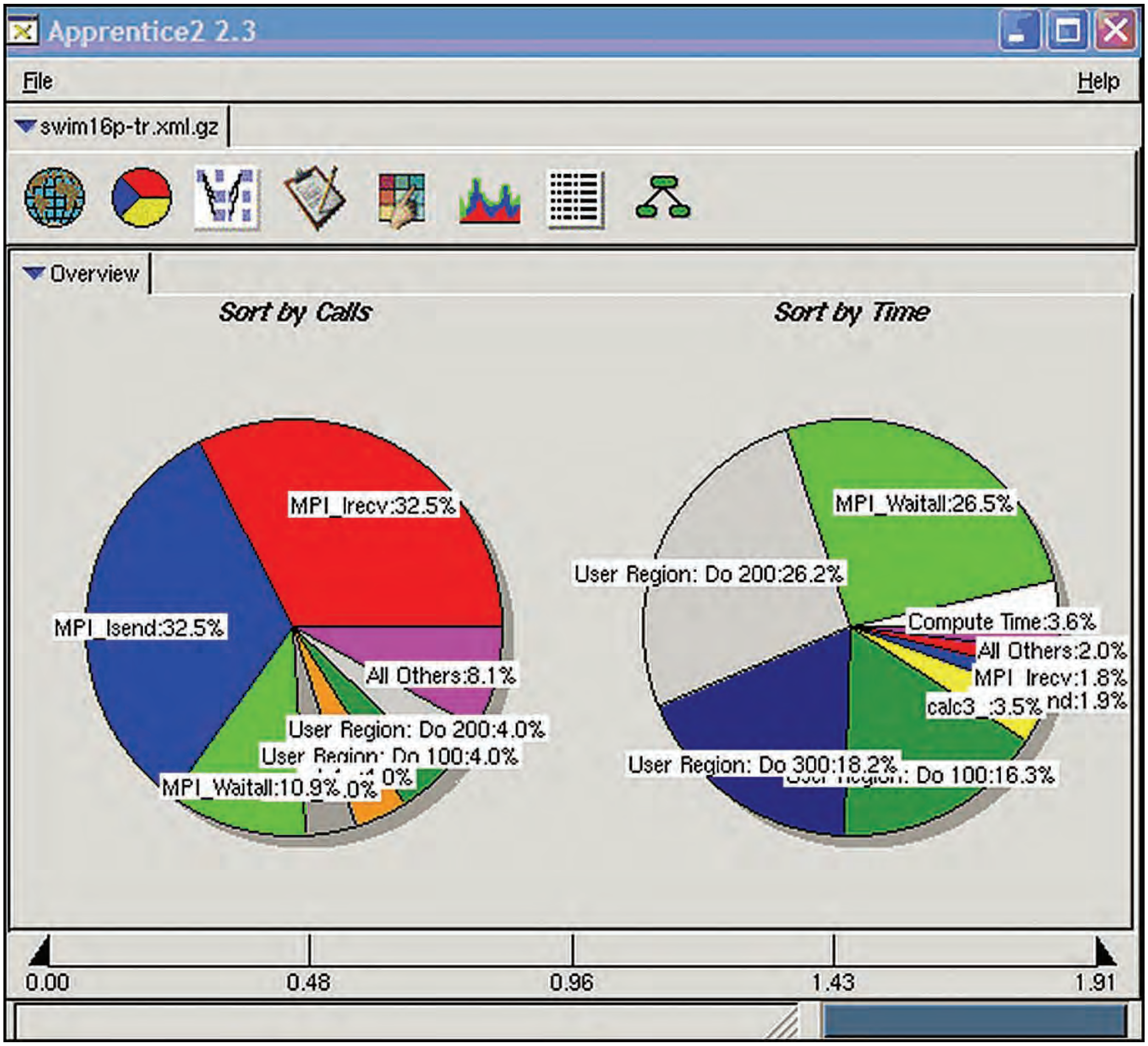


Figure 1. An example of performance data displayed by Apprentice2 on the XT3. Shown are pie charts of percentage of calls made and time spent in the subroutines of the instrumented code. Other displays, such as call trees, are available as well

Cray XT3 Input/Output

By Bobby Hunter and Dr. Jeff Hensley, ERDC MSRC

A key element in any large-scale HPC system is the input/output (I/O) subsystem. The I/O subsystem of the 4,176-processor Cray XT3 at the ERDC MSRC consists of data RAIDs (redundant arrays of independent disks) connected directly to I/O processing elements (PEs) that reside on the high-speed interconnect. The Lustre file system manages the file operations across these RAIDs and offers the users a scalable, secure, robust, highly available cluster file system. In addition, users have available to them a number of options that offer the opportunity to tune the I/O subsystem for their application.

The I/O system hardware uses 23 Opteron-based servers that serve 46 object storage targets (OSTs). Each OST services a raid array of 790 GB, bringing the current total system storage to 36 TB (the total storage space is scheduled to be increased in the near future). Metadata is managed by two service nodes with one node always active and the second used as a failover.

Lustre is a global parallel file system developed by Cluster File Systems in close working relationship with Cray. Lustre stripes the user data across the OSTs. The default number of stripes is 2, and the default stripe size is 1 MB. This means that if a user writes a 2-MB file, 1 MB will be on one physical disk array and the other 1 MB will be on another disk array. By default, the OSTs are selected by the system at file creation time to balance the storage across all of the OSTs. Lustre provides a mechanism that allows the user to specify the OSTs on which a file is to reside. However, this is strongly discouraged since the user may target an OST that may be full or nearly

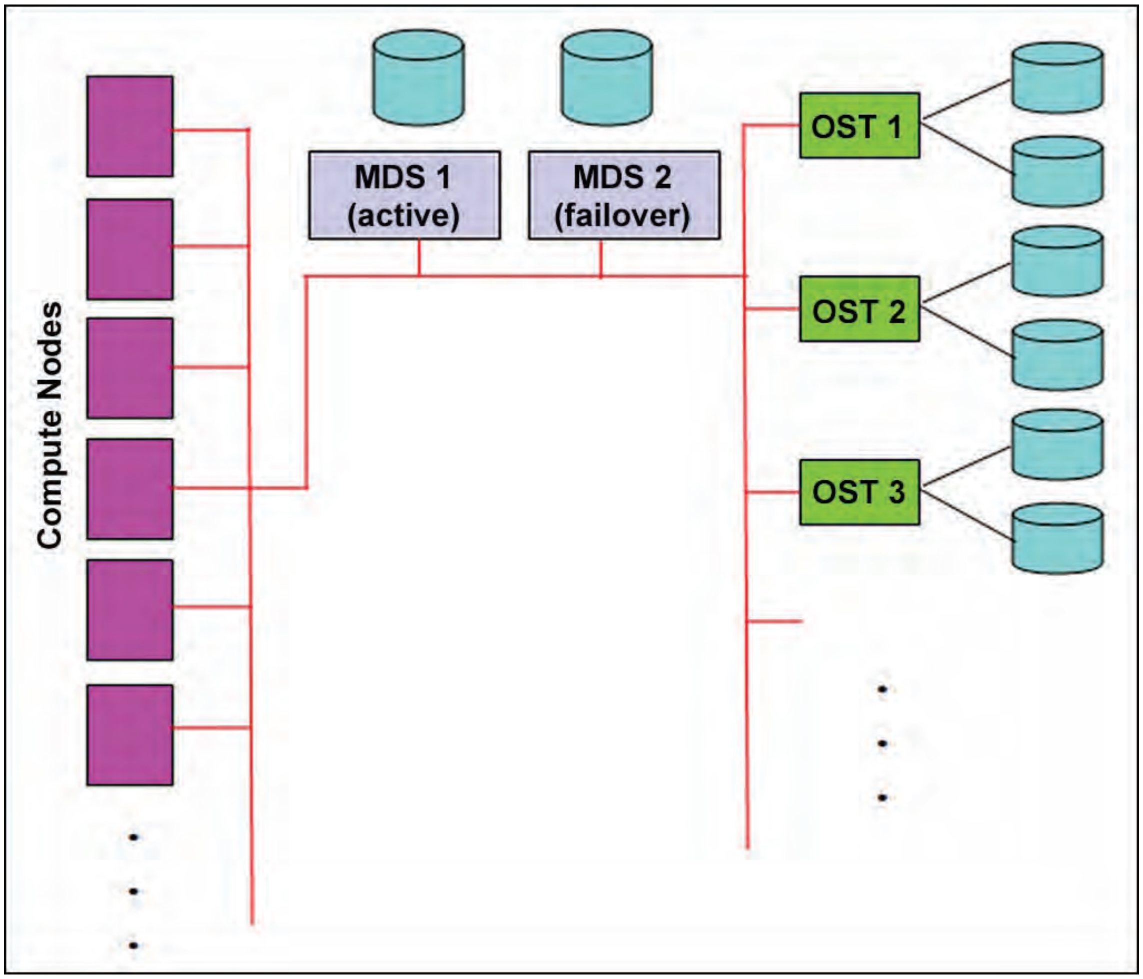


Figure 1. Lustre file system

full. Lustre settings can be changed through the use of the Lustre command **lfs**. The following I/O parameters can be set through **lfs**. The list of parameters is given by the command **lfs help**:

setstripe	find	getstripe
check	catinfo	osts
quotachog	quotacheck	quotaon
quotaoff	setquota	quota

To find out more information on each parameter, use **lfs help** <*parameter name*>.

To change the default striping, use the **setstripe** parameter. The syntax is

```
lfs setstripe  <filename|dirname> <stripe size> <stripe start> <stripe count>
```

where *stripe size* is the number of bytes in each stripe, *stripe start* is the OST index of the first stripe, and *stripe count* is the number of stripes to use. Note that *stripe size* must be in bytes and must be a multiple of 65,536 (64 KB). If the striping is set on a directory, the files created in that directory will inherit the directory's settings by default. For example, the command

```
lfs setstripe <myfile> 131072 -1 8
```

will cause *myfile* to have 8 stripes with a stripe size of 131,072 bytes (128 KB). The argument of -1 indicates the use of the default value for stripe start (it is recommended to let the system choose which OSTs to use). To obtain best performance, this code should write files with a record length of 1 MB (8*128KB).

Striping cannot be changed on an existing file, only on existing directories. When the stripe configuration is changed on a directory, only files created after the change will inherit the new stripe properties. Existing files will remain unchanged.

Listed below are some important points to remember when running applications on the XT3:

1. Remember to write output in `$WORKDIR`.

   You should always run in the `/work` directory. This is the mount point for the Lustre file system while the `/u` directory is NFS-mounted. Users running applications from the `/u` directory will experience degraded I/O performance.

2. Writes to `stdout` and `stderr` can be very slow.

   The bandwidth for writing to `stderr` and `stdout` can be on the order of tens of *bytes* per second. The performance can be improved through the use of the IOBUF library. To use the **xt-iobuf** library, first load the module with

   ```
   module load xt-iobuf
   ```

   Then compile (or relink) your code. In the run script, set the `IOBUF_PARAMS` environment variable

   ```
   setenv IOBUF_PARAMS '%stdout' (csh syntax)
   ```

   which, in this example, will cause stdout to be buffered. It is possible to buffer the output to any file. For more details, see the man page for **iobuf**.

   It is recommended that the user not buffer `stderr`. Buffering of `stderr` could result in the user not receiving error messages if the program terminates abnormally. In general, the user should avoid writing to `stderr` except in the case of a program error.

3. Avoid small writes to the file system.

   When possible have your application buffer as much data as possible before writing to disk. Benchmark tests indicate very poor I/O bandwidth when files are written using small data chunks. IOBUF can be used to increase the effective record length of a program without modifying the code. IOBUF does I/O with a record length equal to the size of a buffer, and thus can be effective for improving performance.

Porting Codes to Sapphire

By Robert Alter, ERDC MSRC

Sapphire is a Cray XT3 Massive Parallel Processor (MPP) with more than 4,000 compute nodes and fully distributed memory. Each compute node consists of a single 2.6-GHz AMD Opteron CPU and is supplied with 2 GB of memory. Porting codes to a new HPC architecture is sometimes challenging, so it is hoped that some helpful information can be found in the following discussion. The parallel programming models available on Sapphire are MPI and SHMEM. Since there is no shared memory between the CPUs, OpenMP is not available.

Two operating systems exist on Sapphire. The login nodes run SuSE Linux, while the compute nodes run Catamount, a microkernel operating system developed by Sandia Laboratories for their XT3 named "Red Storm." Catamount is designed to run one process at a time. It is designed to have a very small footprint in memory and thus has some limitations. User codes intended for the compute nodes cannot use multithreading (pthreads), rpc library calls, X library calls, TCP/IP calls (e.g., **pipe**, **socket**, or IP function calls), or multiple processes (e.g., **fork**, **exec**, or **system** calls). The libraries and include files for these functions have been omitted from the default compiler paths since the Catamount kernel cannot perform these functions.

Sapphire has two compiler vendors, Portland Group, Inc. (PGI), and GNU, with the PGI compilers being the default. The PGI compilers can generate executables for FORTRAN 77/90/95, C, and C++ source files using typical include files and libraries except for the above-mentioned limitations on the compute nodes. There are separate compiler drivers available for running on login or compute nodes because of their different operating systems and the library limitations of the compute nodes. For the compute nodes, the compiler names are **ftn**, **f77**, **cc**, and **CC** for FORTRAN90, FORTRAN77, C, and C++ source files, respectively. For execution on the login nodes, the corresponding compilers are named **linux-pgf90**, **linux-pgf77**, **linux-pgcc**, and **linux-pgCC**. Module commands are used to change compiler versions or change the compiler vendor. The compute node compilers (**ftn**, **f77**, **cc**, and **CC**) automatically link in AMD's ACML math library, Cray's XT3 LibSci scientific library, and the MPICH MPI library. For SHMEM codes, users will need to add "-lsma" to the link step. Using the compiler drivers whenever possible is recommended to ensure that the correct system libraries are being included in user programs.

GNU code for the compute nodes can also be produced by the Cray compiler drivers (**f77** and **cc**). To generate GNU executables, swap the **PrgEnv-pgi** module for the **PrgEnv-gnu** module and compile with the Cray compiler drivers (**ftn**, **cc**, **CC**). Because of some conflicts in include files, generating **g++** code for the compute nodes is not possible.

The GNU compilers for the compute nodes (**g77**, **gcc**, and **g++**) can produce executables for FORTRAN 77, C, and C++ codes, respectively, but have the same Catamount restrictions as the PGI compilers. The corresponding GNU compilers for the login nodes are **linux-g77**, **linux-gcc**, and **linux-g++**. A GNU FORTRAN 90 compiler will be available in the near future. To generate GNU executables for the login nodes, the user loads the **gcc** module. To generate GNU executables for the compute nodes, the user loads the **gcc-catamount** module.

Example compilation commands for running on the compute nodes using the default PGI compiler are as follows:

For aggressive optimization:

```
ftn -fastsse -o my_exec my_prog.f90
```

For moderate optimization (turns off SSE vector instructions):

```
ftn -fast -Mscalarsse -o my_exec my_prog.f90
```

For debugging (no optimization):

```
ftn -O0 -g -Mnoscalarsse -o my_exec my_prog.f90
```

Any code that is to run on the login nodes must be serial and use minimal resources. The login nodes are utilized by all users and should not be used for production runs. Example compilation commands for running on a login node are as follows:

```
linux-pgf90 -fast -o my_exec my_serial_prog.f
```

or

```
      linux-pgcc   -fast -o my_exec my_serial_prog.c
```

For more information about PGI compiler flags, please consult the **pgf90** or **pgcc** man pages.

An MPI C++ Porting Issue

Suppose a PGI compilation of a C++ MPI code fails with the following error.

```
   Catastrophic error:
     #error directive: "SEEK_SET is #defined but must not be for
     the C++ binding of MPI"
```

This error results from an incompatibility between the C++ include files and the MPI standard header file "`mpi.h`" for the keywords `SEEK_SET`, `SEEK_CUR and SEEK_EOF`. To avoid this error, the user has three options:

1. `#undef` these symbols before including `mpi.h`
2. Change the order of included files; place `mpi.h` before `stdio.h` or `iostream`.
3. If MPI I/O is not needed, set the environment variable `MPICH_IGNORE_CXX_SEEK` to 1.

Timers

The user can choose from a variety of timer functions on the XT3. From Cray's XT3 Programming Environment User's Guide S-2396-13:

- Interval timer. Catamount supports the `setitimer ITIMER_REAL` function. It does not support the `setitimer ITIMER_VIRTUAL`, the `setitimer ITIMER_PROF`, or the `getitimer` functions.
- CPU timers. Catamount supports the `getrusage` and `cpu_time` functions. For C and C++ programs, getrusage returns the current resource usages of either `RUSAGE_SELF or RUSAGE_CHILDREN`. The FORTRAN `cpu_time`(*secs*) intrinsic subroutine returns the processor time, where *secs* is either a 4-byte or 8-byte real quantity. The magnitude of the value returned by `cpu_time` is not necessarily meaningful. `cpu_time` is called before and after a section of code, and the difference between the two times is the CPU time used in the section.
- Elapsed time counter. Use the `dclock` or `MPI_Wtime` functions to calculate elapsed time. The `etime` function is not supported.

  The value returned by `dclock` value rolls over approximately every 14 years and is expected to have an accuracy of 100 nanoseconds on each node.

  Note: The `dclock` function is based on the configured processor frequency, which may vary slightly from the actual frequency. Currently, the clock frequency is not calibrated. Furthermore, the difference between configured and actual frequency may vary slightly from processor to processor. Due to these two factors, the accuracy of the `dclock` function may be off by as much as +/-50 microseconds/second or 4 seconds/day.

  The `MPI_Wtime` function returns the elapsed time. The `MPI_Wtick` function returns the resolution of `MPI_Wtime` in seconds.

Endian of Binary Files

By default, Sapphire's binary files are little-endian and IEEE-compliant. By default, files from our SGI Origin 3900 platforms (Ruby, Sand, and Silicon) and our Cray X1 (Diamond) are IEEE-compliant but are big-endian. Our HP/Compaq SC45 (Emerald) has the same binary file format as Sapphire. The PGI FORTRAN compiler will allow Sapphire to read and write big-endian files by using the option `–byteswapio` (or `–Mbyteswapio`). By using this flag when compiling, all files whether read in or written to will be accessed as big-endian. There is no such flag for the PGI C or C++ or any of the GNU compilers.

Launch of ezHPC

By Scotty Swillie, ERDC MSRC

The ERDC MSRC is pleased to announce the launch of a powerful new user tool – ezHPC. ezHPC has gone from a great idea discussed at Users Group meetings to a reality. This Web-based interface will enhance the HPC experience for all users. From job entry to file management to script generation to data storage – ezHPC offers fast, secure access from any computer with Internet access. Simply fire up a browser, log in at *https://solutionhpc.erdc.hpc.mil/*, and you are working.

In recent *Resource* editions, we've given you a look under the hood of this powerful tool. In this article, we want to focus on what you, the user, can expect from ezHPC. The best way to do this is to let you hear from some of your peers – actual ezHPC users. We interviewed two people who have been using ezHPC since the beginning of the beta test period and believe their comments will give you some insight into what you can expect when you log on and put ezHPC to work for you.

Scotty: When did you first start using ezHPC?

Jennifer: I was introduced to ezHPC when it was first released as a demo project about 8 months ago.

Keith: I was part of the first beta group a little over a year ago.

Scotty: Has it become a part of your workflow?

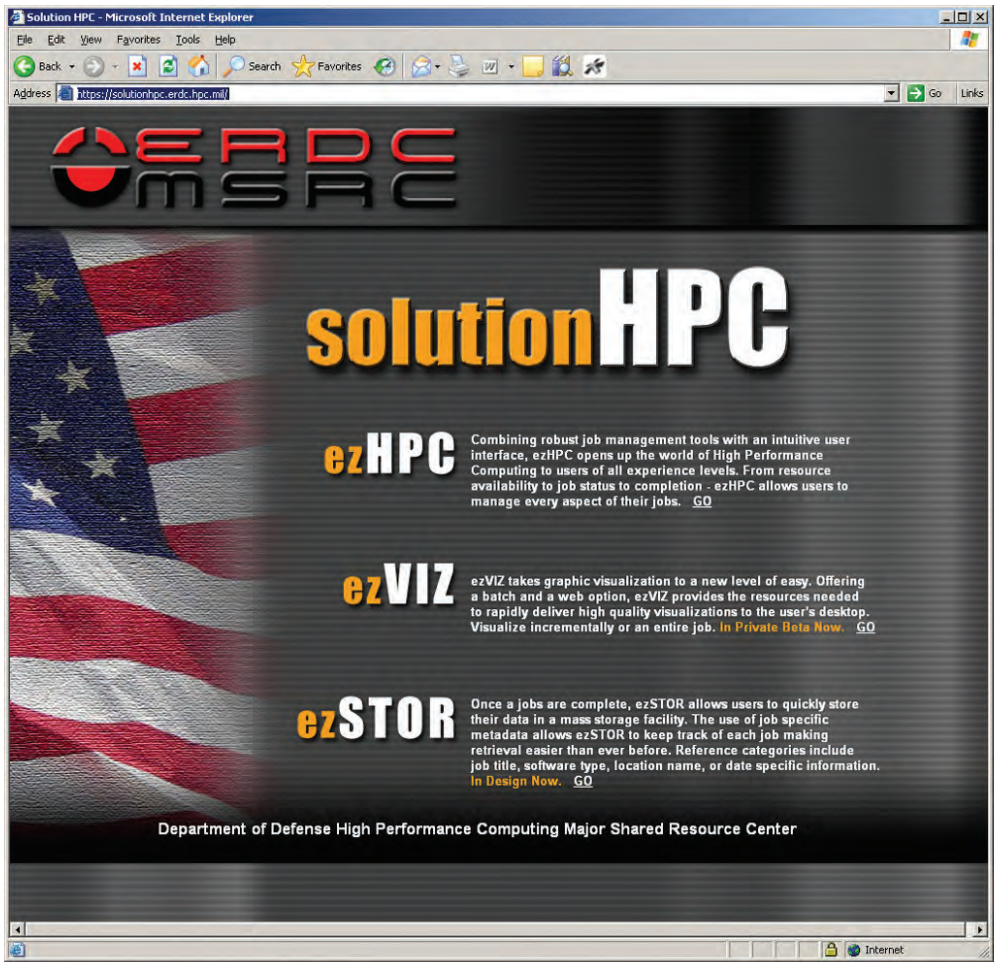
Jennifer: I am gradually working it into my daily workflow - it is a very convenient tool to have available.

Keith: Yes, I use it practically every day. It's been a fairly seamless transition.

Scotty: What kind of computer do you use to access ezHPC, and which browser do you use?

Jennifer: Nothing special - I am using a standard Windows PC with Internet Explorer.

Keith: I use Internet Explorer on a Windows PC and Safari on an Apple G5. Both work without a problem.


Solution HPC - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://solutionhpc.erdc.hpc.mil/
Go Links
ERDC MSRC
solutionHPC
ezHPC
ezVIZ
ezSTOR
Internet

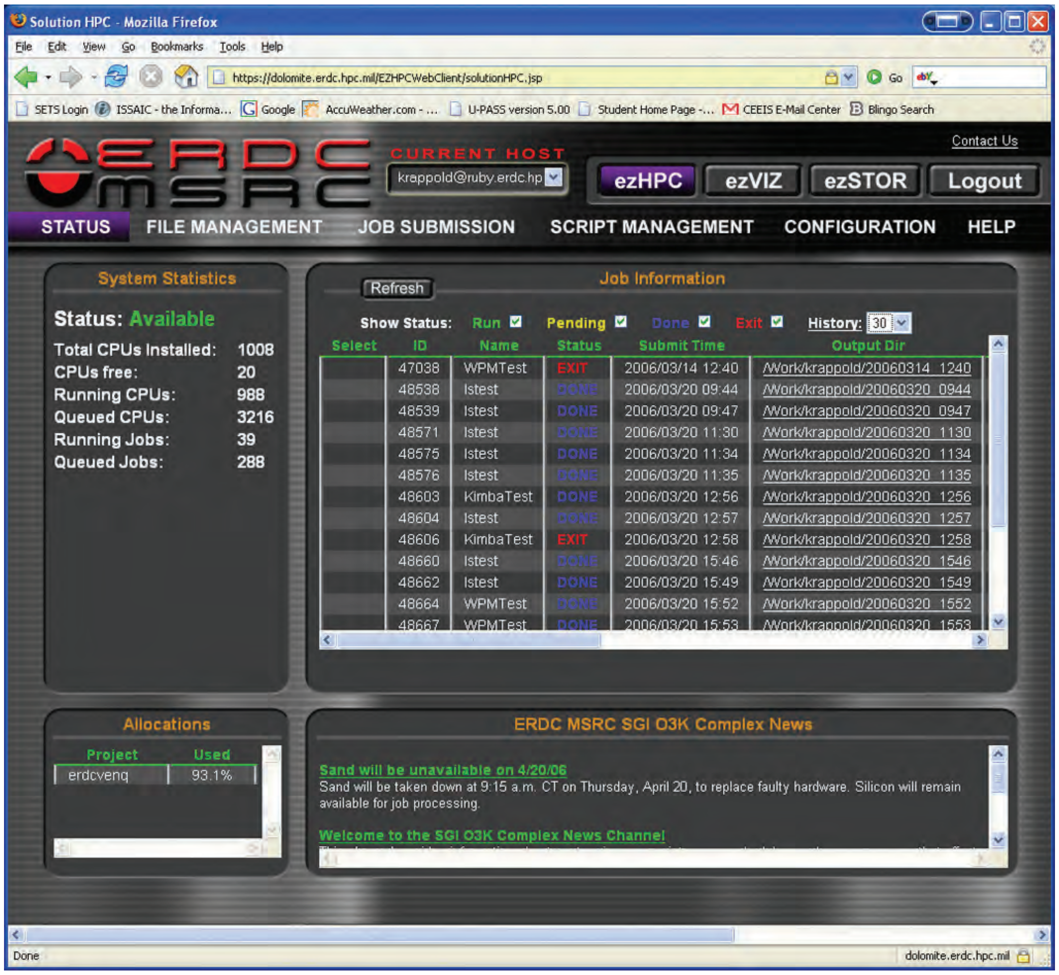
Combining robust job management tools with an intuitive user interface, ezHPC opens up the world of High Performance Computing to users of all experience levels. From resource availability to job status to completion - ezHPC allows users to manage every aspect of their jobs. GO

ezVIZ takes graphic visualization to a new level of easy. Offering a batch and a web option, ezVIZ provides the resources needed to rapidly deliver high quality visualizations to the user's desktop. Visualize incrementally or an entire job. In Private Beta Now. GO

Once a jobs are complete, ezSTOR allows users to quickly store their data in a mass storage facility. The use of job specific metadata allows ezSTOR to keep track of each job making retrieval easier than ever before. Reference categories include job title, software type, location name, or date specific information. In Design Now. GO

Department of Defense High Performance Computing Major Shared Resource Center

Figure 1. solutionHPC introductory screen



System Statistics

Status: Available

Total CPUs Installed:	1008
CPUs free:	20
Running CPUs:	988
Queued CPUs:	3216
Running Jobs:	39
Queued Jobs:	288

Job Information

Show Status: Run ✓ Pending ✓ Done ✓ Exit ✓ History: 30

Select	ID	Name	Status	Submit Time	Output Dir
	47038	WPMTest	EXIT	2006/03/14 12:40	/Work/krappold/20060314_1240
	48538	Istest	DONE	2006/03/20 09:44	/Work/krappold/20060320_0944
	48539	Istest	DONE	2006/03/20 09:47	/Work/krappold/20060320_0947
	48571	Istest	DONE	2006/03/20 11:30	/Work/krappold/20060320_1130
	48575	Istest	DONE	2006/03/20 11:34	/Work/krappold/20060320_1134
	48576	Istest	DONE	2006/03/20 11:35	/Work/krappold/20060320_1135
	48603	KimbaTest	DONE	2006/03/20 12:56	/Work/krappold/20060320_1256
	48604	Istest	DONE	2006/03/20 12:57	/Work/krappold/20060320_1257
	48606	KimbaTest	EXIT	2006/03/20 12:58	/Work/krappold/20060320_1258
	48660	Istest	DONE	2006/03/20 15:46	/Work/krappold/20060320_1546
	48662	Istest	DONE	2006/03/20 15:49	/Work/krappold/20060320_1549
	48664	WPMTest	DONE	2006/03/20 15:52	/Work/krappold/20060320_1552
	48667	WPMTest	DONE	2006/03/20 15:53	/Work/krappold/20060320_1553

Allocations

Project	Used
erdcveng	93.1%

ERDC MSRC SGI O3K Complex News

Sand will be unavailable on 4/20/06
Sand will be taken down at 9:15 a.m. CT on Thursday, April 20, to replace faulty hardware. Silicon will remain available for job processing.

Welcome to the SGI O3K Complex News Channel

Figure 2. ezHPC Job Status Screen

Scotty: Have you tried accessing ezHPC offsite?

Jennifer: Yes, I accessed it at home using a cable modem. I wanted to check on some runs I had loaded. The client worked fine – no issues.

Keith: I have used it at home to check runs and move some files around. I only have dialup, but it worked OK – not as fast as the T1 at work – but it was fine.

Scotty: Has ezHPC been easy to learn?

Jennifer: Yes, and it's been very intuitive for someone used to using a command line. I still use the command line also, but I'm moving more in the ezHPC direction – change is hard.

Keith: Yes. I received a 1-hour primer and that was more than enough to get me up and running. It's intuitive enough that the things I didn't get in the primer, I picked up on quickly.

Scotty: How useful has ezHPC been?

Jennifer: File management - I like ezHPC to transfer between HPC machines as well as to my PC. It is more user-friendly than a command line ftp.

Job submission – I'm getting more comfortable with this part as I use the interface more. I still depend on the command line though. As I said – change is hard.

Keith: File management – It makes it easy to edit my directory structure on various machines and transfer multiple files between those machines.

Job submission – Once I have scripts that are properly configured, it is very easy (no pun intended) to submit the scripts.

Scotty: Have you found it to be an enhancement to your workflow?

Jennifer: Yes. I like being able to easily access the machines from home to check on jobs or make a submittal...that is a definite enhancement to my work since I don't have to make a trip to work at night or on the weekend just to keep things running. To me this is the biggest plus.

Keith: It is more useful in several capacities. Editing, maintaining, and submitting scripts is one area. The things that I said in regard to file management also enhance my workflow.

Scotty: What's your favorite feature?

Jennifer: By far, the file structure and transfer process between HPC machines and the quick view of job status.

Keith: The file management tool is my favorite feature. It really cuts down on those msfgets and msfputs. It's nice to get what I need done in just a few clicks. It's really a step forward.

Scotty: Would you recommend ezHPC to other users?

Jennifer: I would recommend ezHPC to other users who are seeking a faster way to get work done.

Keith: Yes, most definitely. It's really a step forward. I can't wait to see what's next.

To discover how ezHPC can help make your high performance computing life easier - Visit us online at https://solutionhpc.erdc.hpc.mil/.

solutionHPC

User Profiles

Name: Jennifer Tate

Title: Research General Engineer

Organization: ERDC Coastal and Hydraulics Laboratory, Estuarine Engineering Branch

Type of Work: Numerical modeling of hydrodynamics, salinity intrusion, and sediment transport – mostly for Corps Of Engineers Districts (Galveston, New Orleans) and research programs within ERDC

Codes Used: ADH, TABS-MDS

Typical File sizes: Several megabytes to a couple of gigabytes

ERDC MSRC Usage: Runs jobs daily on Ruby (SGI O3K) and Emerald (Compaq SC)

Name: Keith Martin

Title: Research Physicist

Organization: ERDC Coastal and Hydraulics Laboratory, Estuarine Engineering Branch

Type of Work: Numerical modeling of hydrodynamics, salinity intrusion, and sediment transport for Corps Of Engineers Districts and research programs within ERDC

Codes Used: RMA2, SED2D, and RMA10

Typical File Sizes: Usually in the several gigabyte range

ERDC MSRC Usage: Runs jobs daily on Ruby (SGI O3K) and Emerald (Compaq SC)

(Left to right) Dr. Jeffery Holland, Director, Information Technology Laboratory; John E. West, Director, ERDC MSRC; and Dr. Thomas H. Killion, Deputy Assistant Secretary for Research and Technology, and Chief Scientist, Assistant Secretary of the Army Acquisition, Logistics, and Technology, U.S. Army, Washington, D.C., March 23, 2006 (photo courtesy of Rachelle Hinson, ERDC PAO)

John West with Engineer Officers, Contingency Response Unit, Washington D.C., March 31, 2006

John West with Lon Pribble, Congressional Liaison Officer U.S. Army Engineer Research and Development Center, Washington, D.C., January 18, 2006

John West with Engineer Officers, 249th EN BN (Prime Power), Fort Belvoir, Virginia, February 28, 2006

Paul Adams (left), Lead, Scientific Visualization Center, with Lieutenant Colonel William M. Wilson, Provost Marshall's Office, U.S. Army Corps of Engineers, Washington, D.C., January 18, 2006

(Left to right) Ambassador John N. Palmer, former Ambassador to Portugal; Dr. C. Dean Norman, Director, Center for Advanced Vehicular Systems; Leland Speed, Executive Director, Mississippi Development Authority; Hillary Henley, Gulf South Capital; and John West, May 26, 2006

acronyms

Below is a list of acronyms commonly used among the DoD HPC community. These acronyms are used throughout the articles in this newsletter.

ACML	AMD Core Math Library
ADCIRC	Advance Circulation
AFRL	Air Force Research Laboratory
AFSEO	Air Force SEEK EAGLE Office
AMD	Advanced Micro Devices, Inc.
ARL	Army Research Laboratory
ASCE	American Society of Civil Engineers
BLACS	Basic Linear Algebra Communication Subprograms
BLAS	Basic Linear Algebra Subprograms
CAP	Capabilities Application Project
CAT	Computational Aeromechanics Team
CFD	Computational Fluid Dynamics
CPU	Central Processing Unit
CTS	Captive Trajectory System
DoD	Department of Defense
ERDC	U.S. Army Engineer Research and Development Center
FFRM	Finite Fireball Radiation Model
FFT	Fast Fourier Transform
FORTRAN	Formula Translation/Translator (high-level programming language)
GB	Gigabyte
HP	Hewlett-Packard
HPC	High Performance Computing
HPCMP	High Performance Computing Modernization Program
HPM	High-Power Microwave
HSAB	Heavy Storage Adapter Beam
IADS	Integrated Air Defense System
IEEE	Institute of Electrical and Electronics Engineers
I/O	Input/Output
IP	Internet Protocol
IPET	Interagency Performance Evaluation Task Force
ITL	Information Technology Laboratory
KB	Kilobyte
LAPACK	Linear Algebra Package
MALD	Miniature Air Launched Decoy
MB	Megabyte
MPI	Message Passing Interface
MPP	Massively Parallel Processor
MSRC	Major Shared Resource Center
NAVO	Naval Oceanographic Office
NIAB	Non-Ideal Airblast
NFS	Network File System
NRC	National Research Council
OS	Operating System
OST	Object Storage Target
PAO	Public Affairs Office
PE	Processing Element
PET	User Productivity Enhancement and Technology Transfer
PGI	Portland Group, Inc.
RAID	Redundant Arrays of Independent Disk
RISC	Reduced Instruction Set Computing
SAD	Stability Augmentation Device
ScaLAPACK	Scalable LAPACK
SEAD	Suppresion of Enemy Air Defense
SHAMRC	Second-Order Hydrodynamic Automatic Mesh Refinement Code
SHMEM	SHared MEMory
SIMD	Single Instruction, Multiple Data
SSE	Streaming SIMD Extensions
TB	Terabyte
TCP	Transmission Control Protocol
TFLOPS	Trillion Floating-Point Operations per Second

(Continued from page 6)

not only thousands of soldiers, sailors, airmen, and marines to the U.S. Armed Forces, but we play an enormous role in providing those warfighters the tools they need to protect our Nation and a burgeoning research arm to help develop those tools."

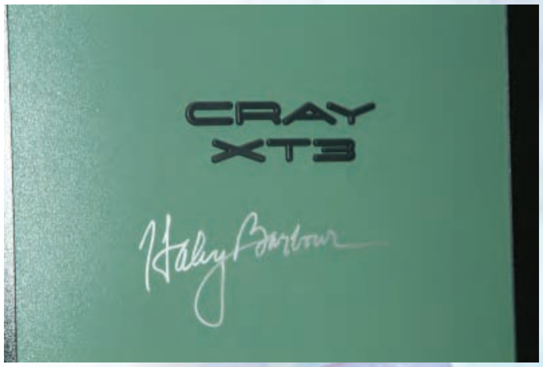
Dr. Jeffery Holland, Director of the ERDC ITL, was Master of Ceremonies for the event as well as serving as host at the ITL facilities. Before a crowd of approximately 150 people, Dr. Holland welcomed special guests including local and State political figures; ERDC and Corps leadership; Cray Henry, Director of the DoD HPCMP; several representatives from Cray, Inc.; as well as ITL personnel. He introduced Dr. James Houston, Director of ERDC, who then introduced the Governor.

A reception followed the Governor's Address, as well as guided tours of the High Performance Computing Center to allow all to view the gleaming spectrum of the supercomputer gems.

22 trillion calculations per second.
One mission.
HIGH PERFORMANCE COMPUTING FOR THE WARFIGHTER

Governor Barbour and Cray Henry, Director, DoD High Performance Computing Modernization Program, unveil portrait of Cray XT3 as Dr. Houston watches

(Left to right) John E. West, Director, ERDC High Performance Computing Major Shared Resource Center; Dr. Houston; Governor Barbour; Cray Henry; Dr. Holland; and Dr. Matt Buckles, Pastor, First Baptist Church, Vicksburg, Mississippi, after unveiling of portrait



Governor Barbour's signature on the Cray XT3

Governor Barbour visits with Professor Joe Thompson (center), Mississippi State University, and Cray Henry at Reception following the Dedication Ceremony

ERDC MSRC *Resource* Editorial Staff

Chief Editor/Technology Transfer Specialist
Rose J. Dykes

Interdisciplinary Coordinator
Mary L. “Dean” Hampton

Visual Information Specialist
Betty Watson

ERDC MSRC HPC Service Center
Web site: *www.erdc.hpc.mil*
E-mail: *msrchelp@erdc.hpc.mil*
Telephone: 1-800-500-4722

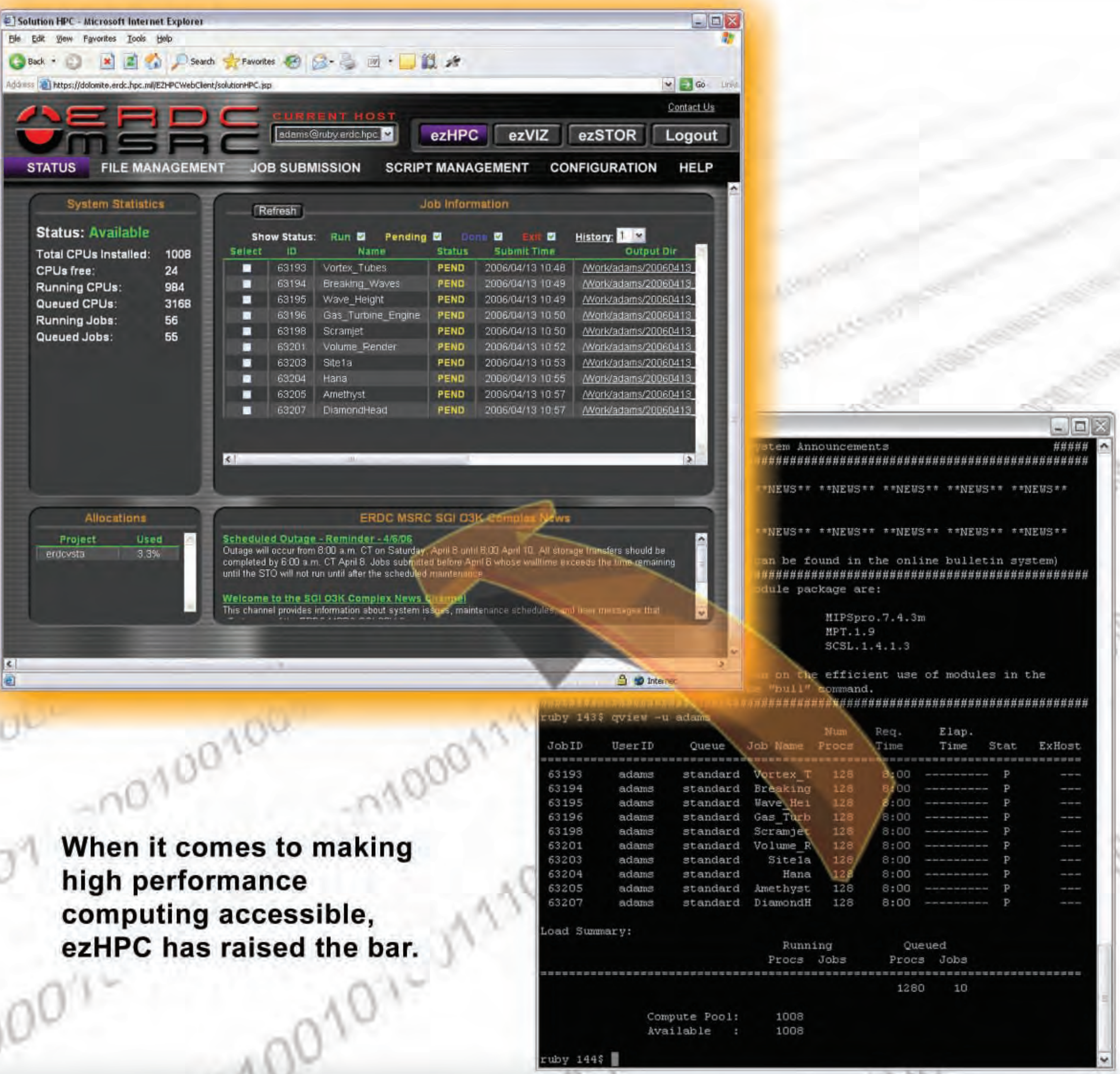
The ERDC MSRC welcomes comments and suggestions regarding the *Resource* and invites article submissions. Please send submissions to the above e-mail address.

Design and layout provided by the Visual Production Center, Information Technology Laboratory, U.S. Army Engineer Research and Development Center.

ezHPC Goes Live!

The rules have changed...

https://solutionhpc.erdc.hpc.mil


Solution HPC - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://dolomite.erdc.hpc.mil/EZHPCWebClient/solutionHPC.jsp
Go Links
ERDC MSRC
Contact Us
CURRENT HOST
adams@ruby.erdc.hpc
ezHPC ezVIZ ezSTOR Logout
STATUS FILE MANAGEMENT JOB SUBMISSION SCRIPT MANAGEMENT CONFIGURATION HELP
Refresh
Internet

System Statistics

Status: Available

Total CPUs Installed:	1008
CPUs free:	24
Running CPUs:	984
Queued CPUs:	3168
Running Jobs:	56
Queued Jobs:	55

Job Information

Show Status: Run ✓ Pending ✓ Done ✓ Exit ✓ History: 1

Select	ID	Name	Status	Submit Time	Output Dir
☐	63193	Vortex_Tubes	PEND	2006/04/13 10:48	/Work/adams/20060413
☐	63194	Breaking_Waves	PEND	2006/04/13 10:49	/Work/adams/20060413
☐	63195	Wave_Height	PEND	2006/04/13 10:49	/Work/adams/20060413
☐	63196	Gas_Turbine_Engine	PEND	2006/04/13 10:50	/Work/adams/20060413
☐	63198	Scramjet	PEND	2006/04/13 10:50	/Work/adams/20060413
☐	63201	Volume_Render	PEND	2006/04/13 10:52	/Work/adams/20060413
☐	63203	Site1a	PEND	2006/04/13 10:53	/Work/adams/20060413
☐	63204	Hana	PEND	2006/04/13 10:55	/Work/adams/20060413
☐	63205	Amethyst	PEND	2006/04/13 10:57	/Work/adams/20060413
☐	63207	DiamondHead	PEND	2006/04/13 10:57	/Work/adams/20060413

Allocations

Project	Used
erdcvsta	3.3%

ERDC MSRC SGI O3K Complex News

Scheduled Outage - Reminder - 4/6/06

Outage will occur from 8:00 a.m. CT on Saturday, April 8 until 6:00 April 10. All storage transfers should be completed by 6:00 a.m. CT April 8. Jobs submitted before April 6 whose walltime exceeds the time remaining until the STO will not run until after the scheduled maintenance.

Welcome to the SGI O3K Complex News Channel

This channel provides information about system issues, maintenance schedules, and user messages that

```
stem Announcements                                  #####
#########################################################

**NEWS** **NEWS** **NEWS** **NEWS** **NEWS**


**NEWS** **NEWS** **NEWS** **NEWS** **NEWS**

an be found in the online bulletin system)
#########################################################
dule package are:

               MIPSpro.7.4.3m
               MPT.1.9
               SCSL.1.4.1.3

   on the efficient use of modules in the
   "bull" command.
#########################################################
ruby 143$ qview -u adams
                                     Num    Req.     Elap.
 JobID    UserID     Queue   Job Name Procs  Time     Time    Stat   ExHost
=========================================================================
 63193     adams    standard Vortex_T  128   8:00  ---------   P       ---
 63194     adams    standard Breaking  128   8:00  ---------   P       ---
 63195     adams    standard Wave_Hei  128   8:00  ---------   P       ---
 63196     adams    standard Gas_Turb  128   8:00  ---------   P       ---
 63198     adams    standard Scramjet  128   8:00  ---------   P       ---
 63201     adams    standard Volume_R  128   8:00  ---------   P       ---
 63203     adams    standard   Site1a  128   8:00  ---------   P       ---
 63204     adams    standard     Hana  128   8:00  ---------   P       ---
 63205     adams    standard Amethyst  128   8:00  ---------   P       ---
 63207     adams    standard DiamondH  128   8:00  ---------   P       ---

Load Summary:
                                   Running          Queued
                                 Procs  Jobs      Procs  Jobs
=========================================================================
                                                  1280    10


               Compute Pool:     1008
               Available   :     1008

ruby 144$ 
```

When it comes to making high performance computing accessible, ezHPC has raised the bar.

U.S. ARMY ENGINEER RESEARCH AND DEVELOPMENT CENTER
INFORMATION TECHNOLOGY LABORATORY